

PROJET FIL ROUGE DATASCIENTEST

# Classification de produits e-commerce Rakuten

Challenge Rakuten France Multimodal Product Data Classification



Antoine PELAMOURGUES

Édouard GRENIER

Hung NGUYEN

Kevin CHENG

Guillaume CLAEYS

Mentor de projet: Francesco MADRISOTTI



DataScientest

---

## TABLE DES MATIÈRES

<b>I. Introduction.....</b>	<b>4</b>
1. Contexte du projet:.....	4
2. Enjeux.....	5
3. Enrichissements d'attributs pour émergence dans les pages de résultats de recherche....	5
4. Objectifs.....	7
<b>II. Exploration des données.....</b>	<b>7</b>
1. Structure générale du dataset.....	8
A. Présentation brute des fichiers.....	8
B. Description des variables.....	9
C. Données manquantes.....	10
D. Données en doublons.....	11
2. Exploration des classes.....	12
A. Feuilletage du catalogue : partie images.....	12
B. Feuilletage du catalogue : partie texte.....	18
a) Étude sur les langues au sein du dataframe.....	19
b) Répartition de la fréquences des mots dans ‘désignation’.....	20
c) Mots les plus fréquents par classe.....	22
d) Proposition de définition des classes.....	23
e) Clustering textuel des classes.....	24
3. Conclusion.....	26
<b>III. Classification unimodale du texte.....</b>	<b>26</b>
1. Pipeline pour la classification texte.....	26
2. Nettoyage et tokenization du texte.....	27
3. Vectorisation par sac de mots.....	30
A. Principe.....	30
B. Implémentation avec Scikit-learn.....	31
C. Conclusion de la phase de preprocessing.....	33
D. Modélisations expérimentées.....	34
E. Résultats.....	36
F. Interprétation avec Lime.....	39
4. Vectorisation par plongement lexical.....	41
A. Principe.....	41
B. Implémentation avec gensim.....	42
C. Résultats et interprétation.....	43
5. Large Language Model.....	43
A. Principe.....	43
B. Architecture et dimension du modèle.....	44
C. Implémentation avec BertForSequenceClassification.....	45

---

---

D. Résultats et interprétation.....	46
6. Conclusion.....	47
<b>IV. Classification unimodale des images.....</b>	<b>47</b>
1. Généralités.....	47
2. Preprocessing des Images.....	48
3. Approche frontale: leNet.....	50
4. Approche Transfer learning.....	50
A. Modèle VGG16.....	52
C. Modèle EfficientNetV2L.....	54
D. Scores obtenus.....	55
5. Interprétabilité via LIME.....	59
A. Interprétation 1: interprétation multiple.....	60
B. Interprétation 2: interprétation correcte.....	61
6. Interprétabilité via Grad-CAM.....	61
7. Conclusion.....	62
<b>V. Classification multimodale des images.....</b>	<b>63</b>
1. Généralités.....	63
2. Nos différentes approches.....	65
A. Fusion précoce.....	65
a) Bert +LSTM (texte) + InceptionV3 (image).....	66
b) LSTM (texte) + EfficientNetB4 (image).....	69
B. Score level fusion.....	70
C. Résultats obtenus.....	72
3. Conclusion.....	74
<b>VI. Création d'une API.....</b>	<b>75</b>
<b>VII. Conclusion.....</b>	<b>77</b>

---

# I. Introduction

Dans le cadre de la formation « Data Scientist », nous avons choisi comme projet le défi « Rakuten France Multimodal Product Data Classification » proposé par le Rakuten Institute of Technology, département de recherche et développement de Rakuten. Notre objectif sera de classer automatiquement des produits en fonction de leurs caractéristiques textuelles et visuelles afin d'améliorer les capacités de recommandation et de recherche personnalisées offertes aux clients des sites e-commerce. Plus précisément, nous chercherons à prédire le code type de chaque produit en utilisant ses descriptions textuelles et son image associée. Pour ce faire, nous collecterons et explorerons des données avant de procéder au prétraitement nécessaire. Ensuite, après avoir choisi une stratégie de modélisation appropriée, nous entraînerons nos modèles et effectuerons une évaluation complète de leurs performances. Nous présenterons également nos interprétations concernant les résultats obtenus. Le reste de ce document expose plus en détail chacune des étapes entreprises.

## 1. Contexte du projet:

L'apparition des nouvelles technologies a révolutionné le secteur de la vente avec l'essor du e-commerce. Ce nouveau marché abolit les anciens carcans de temps et d'espace en permettant la mise en relation immédiate et continue d'acheteurs et de revendeurs sur des distances toujours plus vastes, et cela que ces acteurs soient des professionnels ou encore des particuliers. L'émergence puis l'explosion des ventes en ligne s'inscrit donc naturellement dans le phénomène désigné « Big Data ». Rakuten, une société japonaise créée en février 1997, est l'un des leaders en ce domaine. Elle détient notamment en 2017 le plus grand site de e-commerce du Japon. La plateforme Rakuten compte 1,5 milliard de membres. En 2023, son catalogue d'offres de seconde main est estimé à 50 millions de produits référencés. Les ressources du projet peuvent être consultées à l'adresse suivante : <https://challengedata.ens.fr/challenges/35>. Le sujet consiste à implémenter une solution de classification automatique de produits provenant du catalogue de la plateforme selon la taxonomie propre à cette dernière.

---

## **2. Enjeux**

Ce type de problématique est récurrent en e-commerce. Il est en effet vital pour une plateforme en ligne de garantir un catalogage conforme de ses produits car en découlent la performance et l'optimisation de ses services commerciaux : recommandations et visibilité de produit, recherches personnalisées, compréhension des requêtes des utilisateurs. Bien que nécessaire, la résolution de ces sujets est difficilement généralisable en une approche unique. Tout d'abord car chaque plateforme dispose d'un éventail de produits et d'une taxonomie qui lui sont propres. Certaines, dont Rakuten, comportent de plus une part importante de leurs revendeurs qui ne sont pas des professionnels, ce qui impacte l'étendue et la qualité des données.

## **3. Enrichissements d'attributs pour émergence dans les pages de résultats de recherche**

Les attributs enrichis par les e-marchands sur les marketplaces leur permettent de gagner en visibilité. Cette visibilité peut être relative aux pages de résultat sur leur propre moteur de recherche interne, plus utilisé pour les sites de vente de seconde vie (ex: Le bon coin) ou bien sur les moteurs de recherche tels que Google ou encore Bing (ex: Etsy, ebay, Rakuten). Cette visibilité est clé car elle permettra aux sites e-commerce de générer plus de trafic et donc de gagner plus de parts de marché.

Dans le premier cas, l'impact peut être moins important car la priorisation sur les pages de résultats de recherche dépendent des règles de merchandising mises en place par le site e-commerce. Pour une recherche sur le bon coin, à moins de payer pour que l'annonce soit en visibilité, le site affiche les résultats en effectuant un tri décroissant par rapport à la date de publication de l'annonce.

Dans le second cas, pour que les produits puissent être diffusés sur les moteurs de recherche sur la partie shopping, les e-commerçant utilisent des flux. Ces flux sont constitués d'un certain nombre d'attributs, certains obligatoires (avec notamment le titre, la description et le lien image), d'autres comme recommandés qui sont renseignés par les e-marchands quand ils complètent leurs fiches produit. Les attributs obligatoires dépendent de la typologie de produits. Ces champs permettent à Google de catégoriser les produits pour répondre aux requêtes

utilisateurs en affichant les produits les plus appropriés. Par exemple, il existe de nombreux attributs obligatoires pour la catégorie Vêtements et accessoires, notamment la couleur, le sexe ou encore la tranche d'âge. Google recommande l'enrichissement d'une vingtaine d'attributs. Il est nécessaire d'enrichir un maximum d'attributs dans les flux produits (renseignés par les e-marchands). L'enrichissement d'attribut permettra de ressortir dans d'avantages de recherches utilisateurs. Ce travail permettra de répondre à un éventail plus large de requêtes, mais aussi dans les recherches dites "longue traîne" (ou requêtes longues, c'est-à-dire des requêtes constituées de plus de 5 mots) qui sont de plus en plus fréquentes ces dernières années.

### Exemple sur la requête "canape droit 3 places"

La moteur de recherche propose un certain nombre de canapés que nous pouvons retrouver ci-dessous.

Section	Produit	Prix	Magasin	Référence
Sponsorié	IKEA - VIMLE housse canapé... 209,00 € IKEA	209,00 €	IKEA	IKEA - VIMLE housse canapé 3 pl. Gunnared beige
	IKEA - VIMLE housse canapé 3 pl. Gunnared beige 209,00 € IKEA	209,00 €	IKEA	
Canapés modulables	DRIVE Canapé 3 places GORDON tissu... 399,99 € But	399,99 €	But	DRIVE Canapé 3 places GORDON tissu...
	DRIVE Canapé 3 places GORDON tissu... 399,99 € But	399,99 €	But	
Canapés convertibles	IKEA - Coussins - VIMLE canapé ... 649,00 € IKEA	649,00 €	IKEA	IKEA - Coussins - VIMLE canapé ...
	IKEA - Coussins - VIMLE canapé ... 649,00 € IKEA	649,00 €	IKEA	
Tissu	DRIVE Canapé 3 places COAST II tissu... 499,00 € But	499,00 €	But	DRIVE Canapé 3 places COAST II tissu...
	DRIVE Canapé 3 places COAST II tissu... 499,00 € But	499,00 €	But	
Canapés inclinables	Canapé 3 places beige style... 899,00 € Maisons du Mo...	899,00 €	Maisons du Mo...	Canapé 3 places beige style...
	Canapé 3 places beige style... 899,00 € Maisons du Mo...	899,00 €	Maisons du Mo...	
Causeuses	PROMOTION Canapé 3 places LUKI tissu Asto... 499,00 € 629,-€ But	499,00 € 629,-€	But	PROMOTION Canapé 3 places LUKI tissu Asto...
	PROMOTION Canapé 3 places LUKI tissu Asto... 499,00 € 629,-€ But	499,00 € 629,-€	But	
Cuir	Par Keyade		Par Keyade	Par Keyade
	Par Keyade		Par Keyade	
Dossier fixe	Par Yteo		Par Yteo	Par Yteo
	Par Yteo		Par Yteo	

Nous réalisons maintenant une nouvelle requête "canape droit 3 places tissu beige".

Le moteur de recherche nous propose la page de résultats ci-dessous. Le canapé "IKEA - VIMLE housse canapé 3 pl. Guannared beige" proposé sur la première page de résultat n'est pas présent. En regardant le titre dans son intégralité, l'attribut relatif à la matière (tissu) n'est pas présent. Il est possible qu'il ne soit pas renseigné dans les attributs du produit par l'e-commerçant, le privant de visibilité et donc de potentiels acheteurs.

The screenshot shows a search results page for "canape droit 3 places tissu beige". The results are labeled as "Sponsorié". There are eight items displayed, each with a small image, the product name, price, and a brief description. The items include:

- Canapé 3 places en tissu texturé beige SOPELANA - 699,99 €
- Canapé en tissu 3 places écrue Lars Tikamoon - 799,00 €
- PROMOTION Canapé convertible 3 places BOLERO tissu bouclette... - 699,99 €
- Canapé droit tissu chenille "Matignon" - 3 places - Beige - 599,00 €
- Canapé droit fixe ultra moelleux MONT-BLANC tissu chiné beige... - 849,00 €
- sweeek Canapé 3 places cosy rond, tissu bouclettes... - 499,99 €
- PROMOTION Canapé 3 Places Blanc Cassé en Tissu Chenille avec Coussins... - 679,99 €
- Regala canapé 3 places convertible avec coffre en tissu... - 399,00 €

## 4. Objectifs

Ce projet est ambitieux : il oriente dès son énoncé à recourir à des méthodes complexes de Deep Learning ce que nous ferons plus tard. Il s'agit en effet de battre un modèle Benchmark qui traite séparément images et textes avec deux algorithmes d'apprentissage profond. La métrique d'évaluation utilisée est le F1-score pondéré.

Performance du modèle benchmark	Texte	Images
F1-score pondéré	81.13 %	55.34 %

## II. Exploration des données

Pour les besoins du challenge, Rakuten IT met à notre disposition un vaste jeu de données transcrivant des annonces réelles issues de leur plateforme. Comme nous allons le voir, son contenu est riche.

---

## 1. Structure générale du dataset

### A. Présentation brute des fichiers

Quatre ressources sont délivrées :

- ❖ 3 fichiers au format csv: X\_train, Y\_train, X\_test
- ❖ 1 dossier compressé images.zip contenant toutes les images

Ce découpage en ensembles d'entraînement et de test est la forme mise en place usuellement pour l'évaluation d'un modèle. Cela certainement dans le but de classer les participants au challenge. S'agissant d'un challenge, nous n'avons pas accès au fichier y\_test. Cela rend les données de X\_test difficilement utilisables. Nous faisons donc le choix d'ignorer ce dernier et nous nous référerons à X\_train et Y\_train lorsque nous évoquerons le jeu de données.

Voici un aperçu succinct des dataframes X\_train, X\_test et y\_train:

X\_train (dimension du dataframe : 84916 x 4)

	designation	description	productid	imageid
0	Olivia: Personalisiertes Notizbuch / 150 Seite...	NaN	3804725264	1263597046
1	Journal Des Arts (Le) N° 133 Du 28/09/2001 - L...	NaN	436067568	1008141237
2	Grand Stylet Ergonomique Bleu Gamepad Nintendo... PILOT STYLE Touch Pen de marque Speedlink est ...	201115110	938777978	
3	Peluche Donald - Europe - Disneyland 2000 (Mar...	NaN	50418756	457047496
4	La Guerre Des Tuques	Luc a des id&acute;es de grandeur. Il veut or...	278535884	1077757786

X\_test (dimension du dataframe : 13812x 4)

	designation	description	productid	imageid
84916	Folkmanis Puppets - 2732 - Marionnette Et Théâ...	NaN	516376098	1019294171
84917	Porte Flamme Gaxix - Flamebringer Gaxix - 136/...	NaN	133389013	1274228667
84918	Pompe de filtration Speck Badu 95	NaN	4128438366	1295960357
84919	Robot de piscine électrique	<p>Ce robot de piscine d'un design innovan...	3929899732	1265224052
84920	Hsm Destructeur Securio C16 Coupe Croisé: 4 X...	NaN	152993898	940543690

y\_train (dimension du dataframe : 84916 x 4)

prdtypecode	
0	10
1	2280
2	50

---

## **B. Description des variables**

- ❖ **désignation** : texte libre en langage naturel sans modèle s'apparentant au titre de l'annonce et parfois suivi d'une description;
- ❖ **description** : texte libre en langage naturel sans modèle qui contient une description plus détaillée du produit;
- ❖ **productid et imageid** : des entiers servant de coordonnées identifiant l'image commerciale rattachée au produit dans le dossier images.zip. Les deux nombres sont nécessaires pour retrouver le fichier. Chaque observation est en correspondance unique à une image couleur RGB de taille 500\*500 pixels.

Au travers de cette courte revue transparaît l'ambivalence des features décrivant les observations: à la fois très réduites en quantité et très différentes (grossièrement: une feature texte décomposée en un titre et une éventuelle description, puis une feature image) et en même temps extrêmement riches.

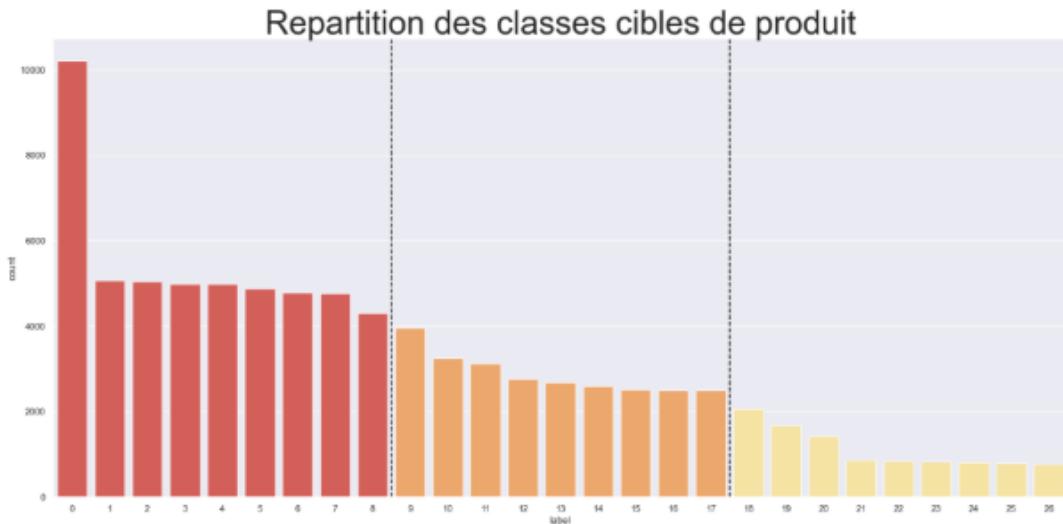
Le dernier fichier y\_train ne contient qu'un champ:

- ❖ **prdtypecode** : entier correspondant à un code produit de la taxonomie Rakuten

Ces codes produits, non définis, sont les étiquettes de notre problème de classification multi-classes. Il y a en tout **27 codes produits différents**. Ce nombre important de classes est une difficulté supplémentaire : plus le nombre de sorties est élevé, plus les possibilités d'erreur sont nombreuses, en particulier pour des classes "proches". Un estimateur trivial qui attribuerait complètement au hasard les résultats aurait une accuracy de 3,7 %, à comparer aux 50 % qu'il obtiendrait dans le cadre d'une classification binaire.

De surcroît, l'absence d'une définition "officielle" nous invite à extraire nous-mêmes le contour de chaque classe, ce qui n'est pas forcément chose aisée comme nous le verrons un petit peu plus tard. Le catalogue complet de produits présents sur Rakuten France est bien entendu nettement plus important que 99k lignes et contient bien plus que 27 codes produit.

La distribution du nombre de produits par type montre une grande disparité de proportion sur les codes produits représentés. La classe majoritaire est environ 10 fois plus présente que la classe minoritaire :



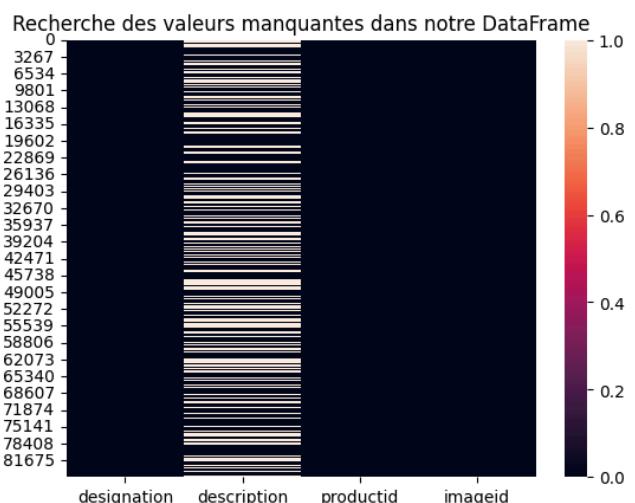
En divisant les 27 labels en 3 batchs ordonnés par fréquence, on observe que :

- ❖ batch rouge: les 9 labels les plus présents, représentent à eux seuls environ 58% du total
- ❖ batch orange: les 9 labels suivants, environ 30%
- ❖ batch jaune: les 9 derniers labels, moins de 12%

Nous sommes donc dans le cadre d'une **classification déséquilibrée**. Nous devons par conséquent prendre garde à l'accuracy paradox : un classifieur trivial qui attribue systématiquement l'étiquette de la classe dominante affichera une accuracy de 12,0% (contre 3,7% pour le classifieur aléatoire). Cela justifie que le f1-score soit la métrique retenue car elle permet de mieux lisser ce type de disparités.

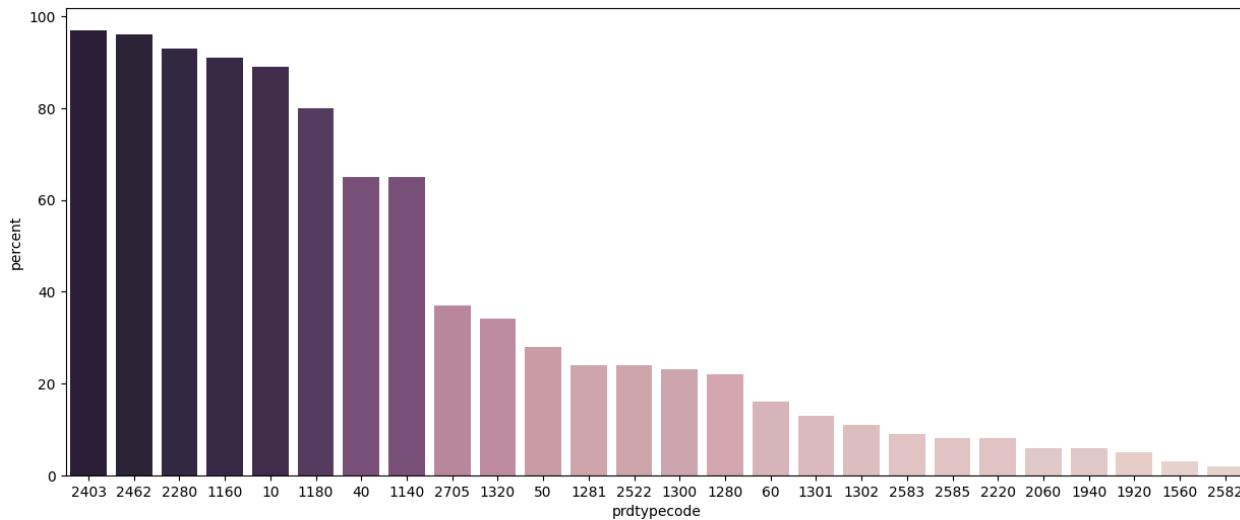
### C. Données manquantes

Comme le montre la carte de chaleur, le seul champ touché par les valeurs manquantes est ‘description’ mais il l'est fortement : environ 35 % de données manquantes !



---

Voici plus en détail comment ces valeurs manquantes se distribuent par labels:



Nous observons de fortes disparités entre les diverses classes : certaines sont quasiment vides quand d'autres sont presque entièrement remplies. Nous émettons l'hypothèse que pour figurer sur la plateforme, il est exigé que le vendeur fournit à minima un titre et une photo, mais que la description soit en revanche facultative. Certains types de produits sont beaucoup plus vendus par des particuliers et nous pouvons imaginer que ces derniers se dispensent plus fréquemment de renseigner une description, estimant que le titre et la photographie suffisent. Les classes les plus vides de descriptions pourraient donc concerner essentiellement des produits d'occasion, voire être définies comme telles.

En terme de modélisation il va falloir définir une stratégie pour le champ description. Faut-il le supprimer, le fusionner au champ "désignation" ou utiliser une méthode d'imputation pour le remplir ?

## D. Données en doublons

Stricto sensu, le dataset ne présente aucun doublon complet ie avec une ligne entièrement identique: chaque ligne correspond bien à une annonce unique. Prises isolément, les colonnes 'désignation' et 'description' présentent respectivement 3 % et 9 % de doublons. Les images présentent également 7% de doublons (5692 images) mais sont bien associées dans la grande majorité des cas à des produits distincts. Le phénomène de « repost » existe, c'est-à-dire que certains annonceurs publient une copie exacte d'une annonce déjà existante, peut être dans le but de la mettre en avant ou encore par mégarde. Nous avons également relevé des produits identiques mais classés de différentes façons :

---

2585: Brico, Jardin > BRICOLAGE



2582: Maison > JARDIN & EXTÉRIEURS



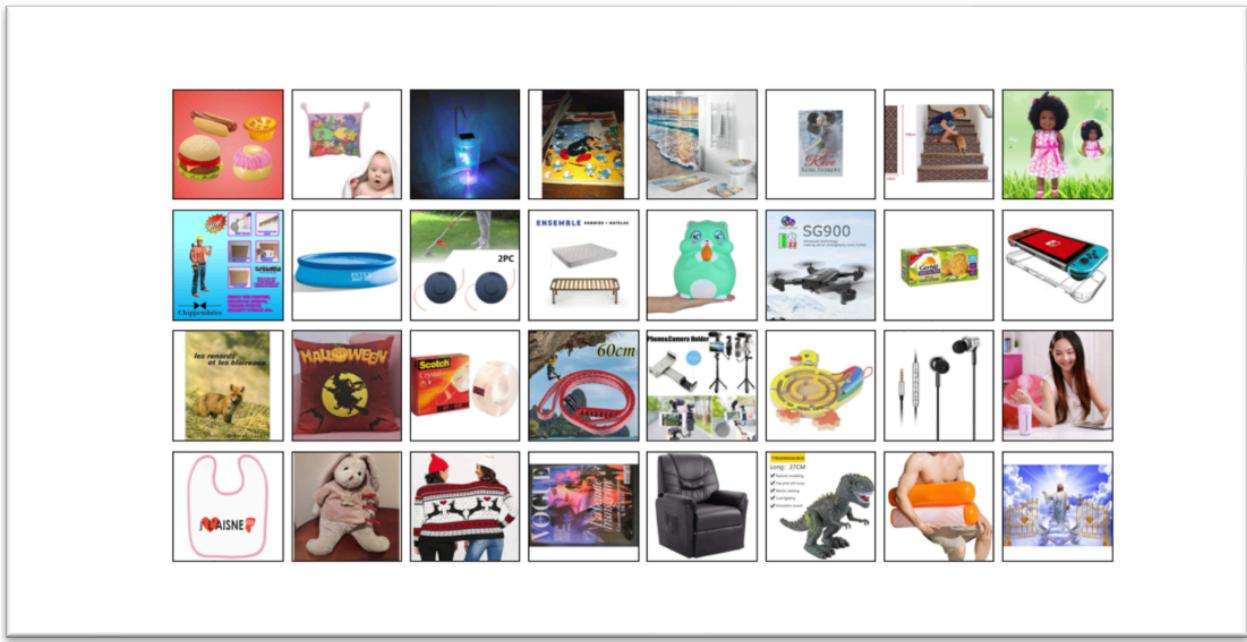
Les utilisateurs étant libres de définir la catégorie de leur produit, il peut apparaître des contradictions du type de celles relevées ci-dessus. Nous pourrions supprimer ou corriger ces données, mais cela reviendrait à arbitrer une classification correcte, or nous ne connaissons pas précisément les principes de la taxonomie Rakuten. Ce type particulier d'incohérence reste relativement rare dans notre dataset, mais il expose le problème plus général de sa data quality, qui va représenter un sérieux obstacle aux performances qui peuvent être atteintes par notre modélisation.

Plutôt que les entrées, nous aurions pu redéfinir les labels en exploitant le clustering en catégories mères, mais ce faisant on aurait appauvri considérablement les nuances présentes dans le dataset tout en s'éloignant de l'objectif métier. Plus simplement, nous étudierons une autre métrique de scoring : la top3 accuracy. Selon ce principe, on n'attend plus forcément d'un modèle qu'il fournisse la prédiction correcte, mais on se satisfait qu'elle se trouve parmi ses trois meilleures suggestions.

## **2. Exploration des classes**

### **A. Feuilletage du catalogue : partie images**

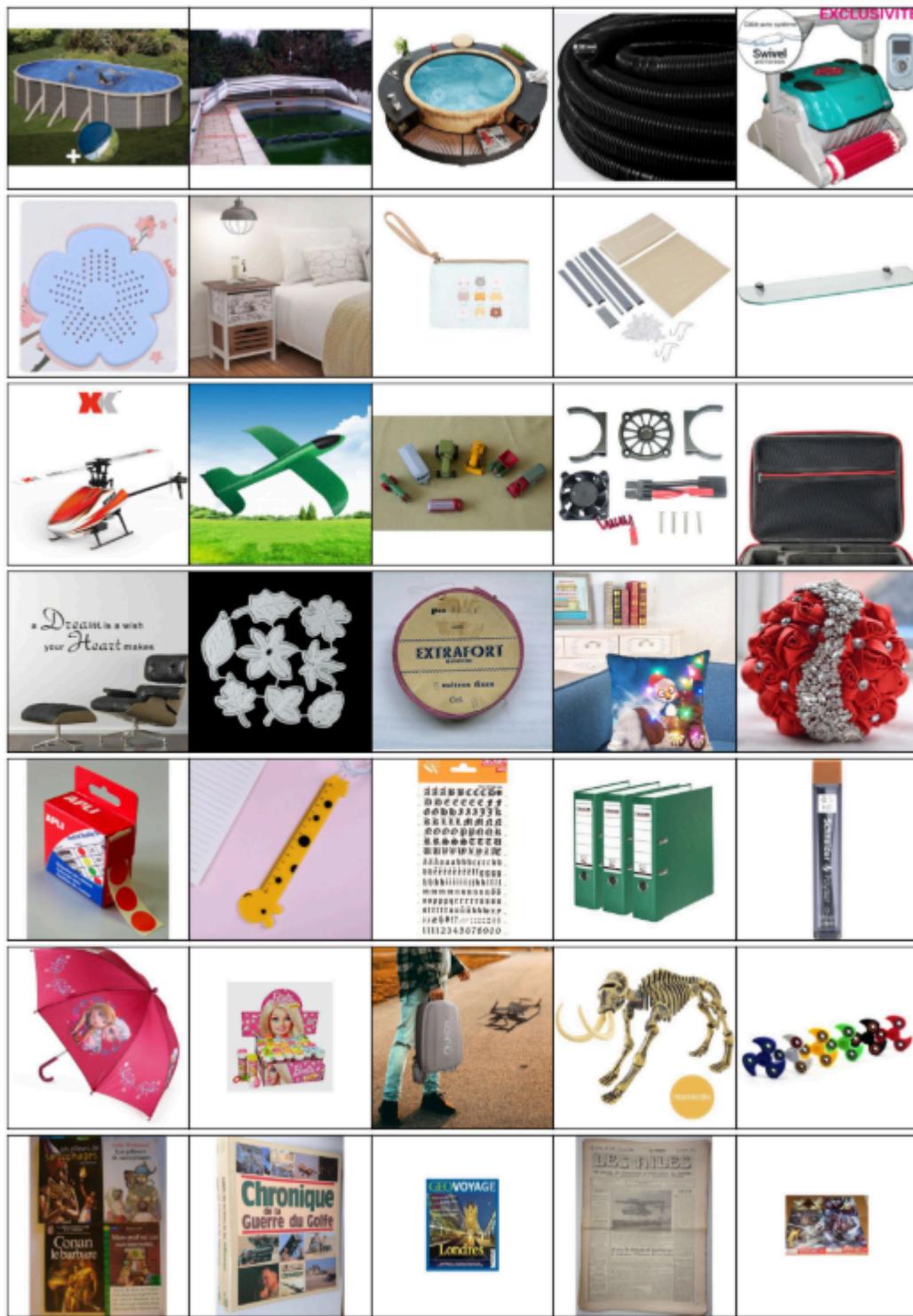
Dans la mesure où notre dataset s'apparente à un catalogue de produits, il semble naturel à ce stade d'en feuilleter les pages pour mieux en cerner les contours, d'autant que les codes produits n'ont pas encore fait l'objet d'une définition.

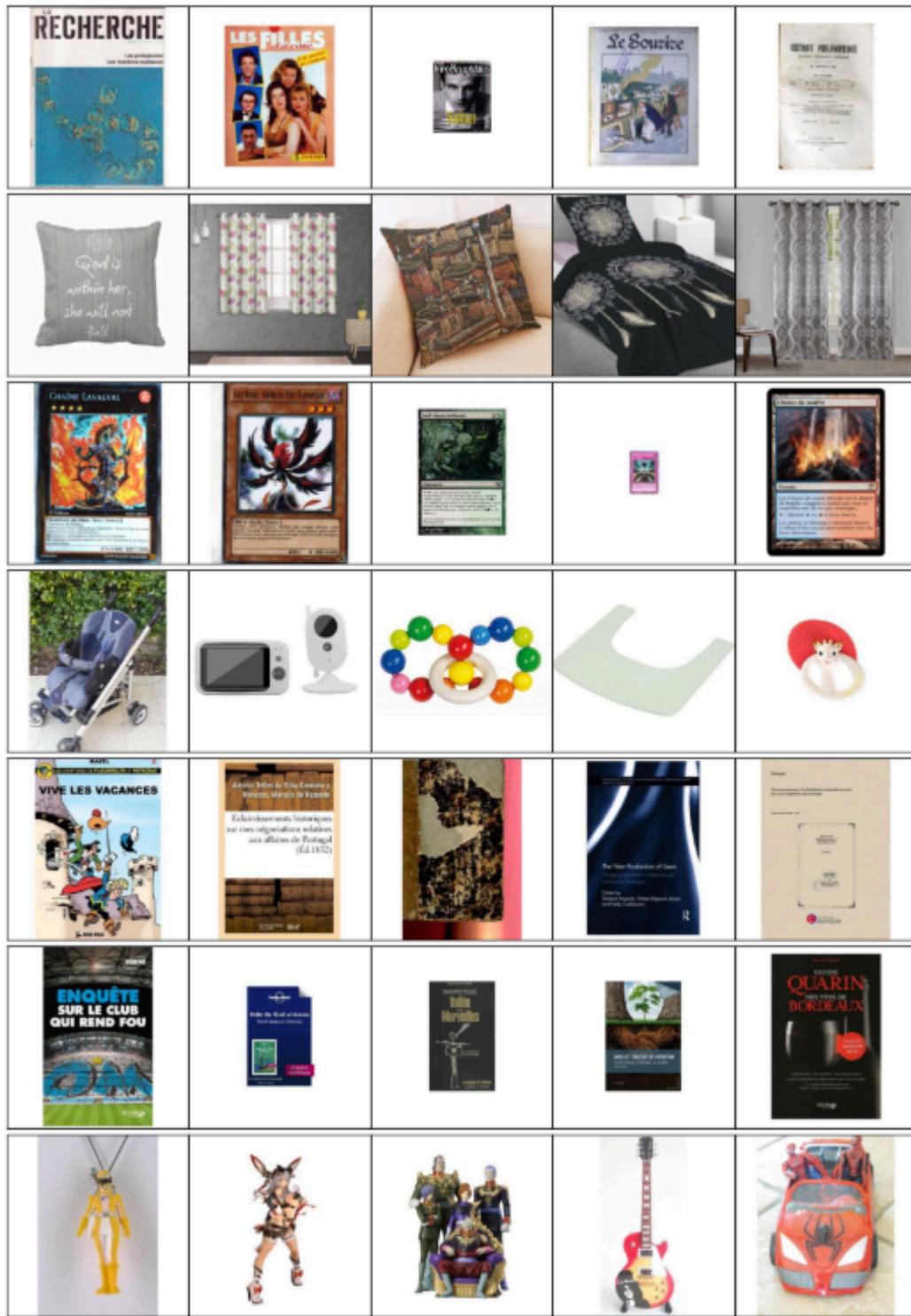


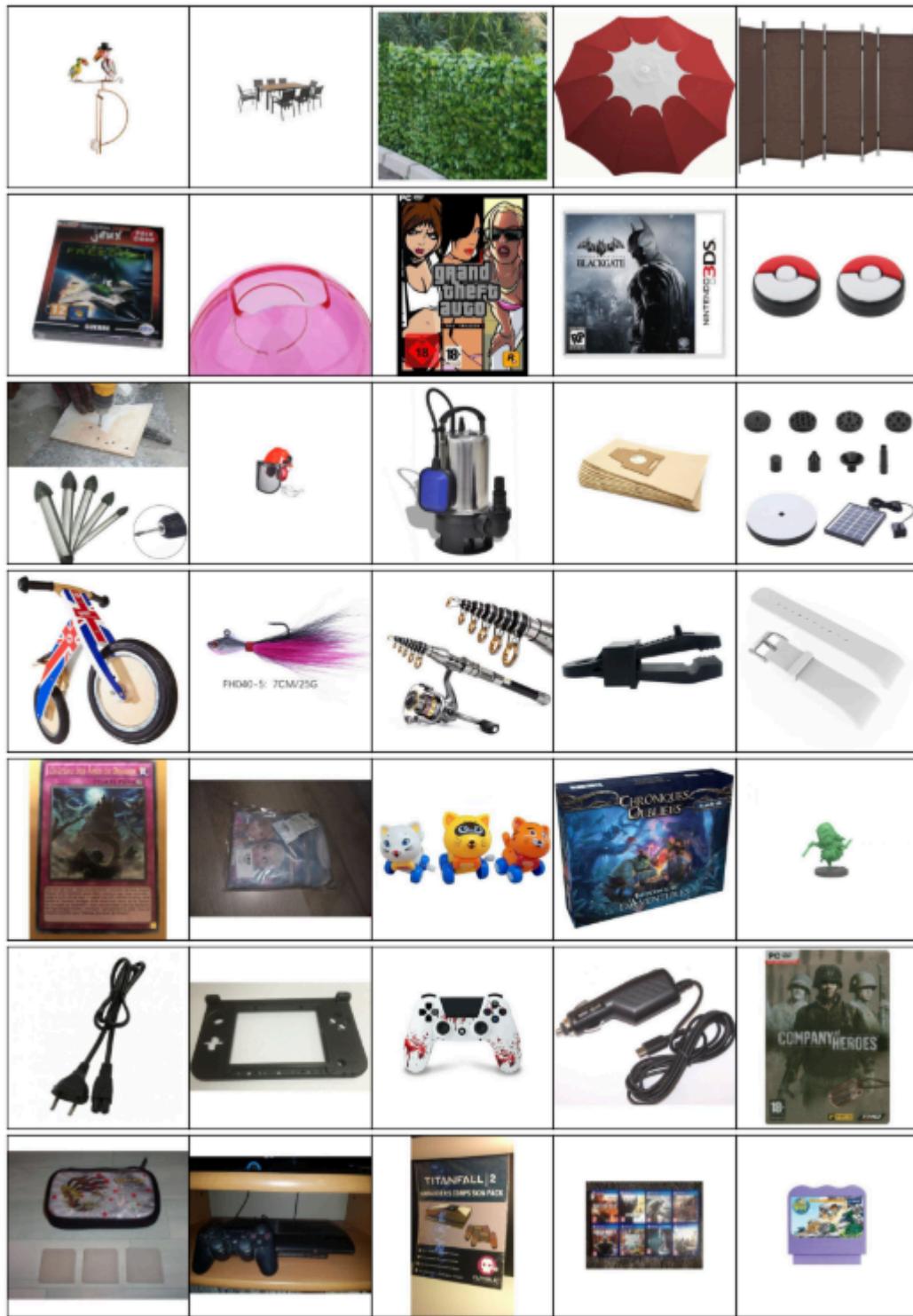
Cette première plongée dans la partie images du dataset nous permet de réaliser la très grande diversité des produits du catalogue, voire même leur originalité. C'est toute la magie des plateformes de e-commerce comme Rakuten qui mettent en relation un nombre tel d'acteurs qu'il devient concevable de trouver preneur pour tous types de biens.

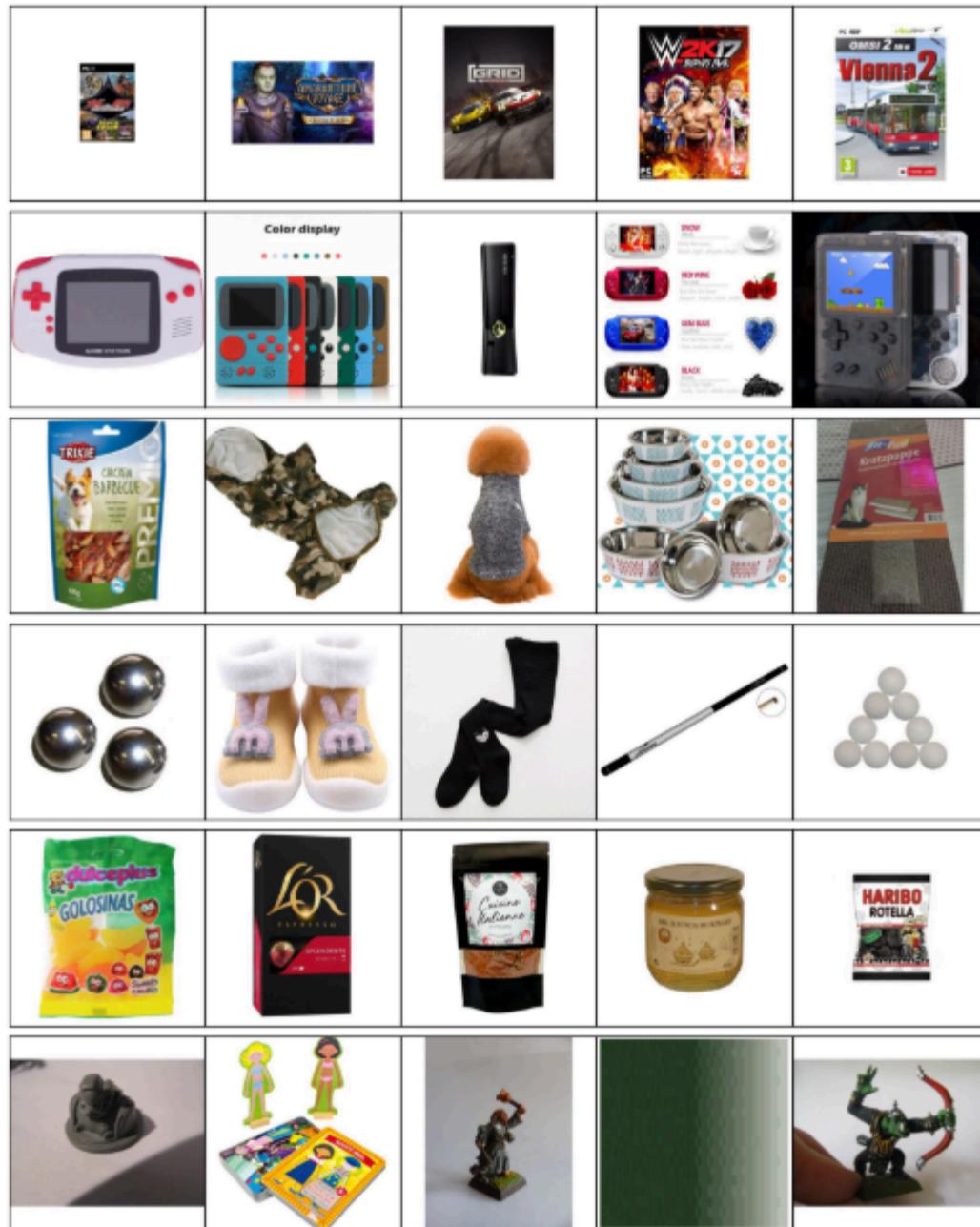
Il est à noter que les images sont loin d'être toutes de même qualité. Ce sont en majorité des photos ou des dessins d'objets 2D ou 3D plus ou moins éloignés, présentant des jeux de couleurs, d'exposition à la lumière, de netteté et de contraste variables. L'image est parfois parasitée par des objets apparaissant dessus mais qui ne désignent pas l'article à vendre.

Voici un affichage d'un échantillon de 5 images pour les 27 classes, chaque ligne représente un code-produit différent et ces lignes sont classées par ordre décroissant d'importance de fréquence :









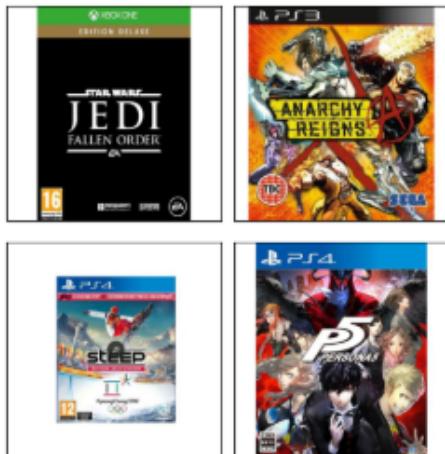
Si le périmètre de certaines classes apparaît naturellement, cela est moins évident pour d'autres. Voici un exemple du code produit 1301 que nous définissons comme générique :



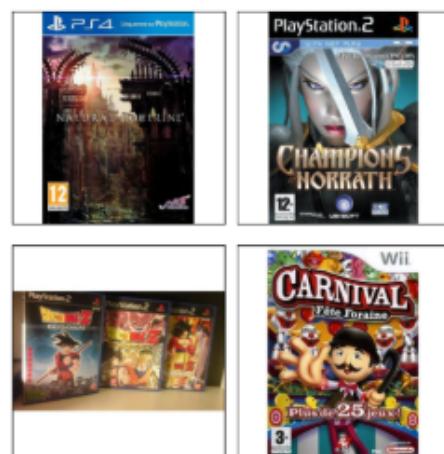
---

Nous sommes également face à des codes produits qui semblent difficilement discernables:

Code produit 40



Code produit 2462



## B. Feuilletage du catalogue : partie texte

Voici un exemple de 5 textes au sein de la colonne ‘désignation’:

1. Dishonored: Dunwall City Trials (Extension Dlc) - Jeu En Téléchargement
2. Lewsor 60pv Destinées Futures 61/99
3. Pierre Renee Gosset Tome 1 Et 2 Adolf Hitler (( De La Prise Du Pouvoir A Munich
4. Chaîne de tronçonneuse OZAKI semi carrée sous coque: 3/8LP - .043 (11mm) - 33 entraîneurs - Longueur : 8" / 20 cm
5. 1 Pc Cute Cartoon Voyage Portable Protection Brosse À Dents Head Hôtel Salle De Bainshpp2004

Voici un exemple de 4 textes au sein de la colonne ‘description’:

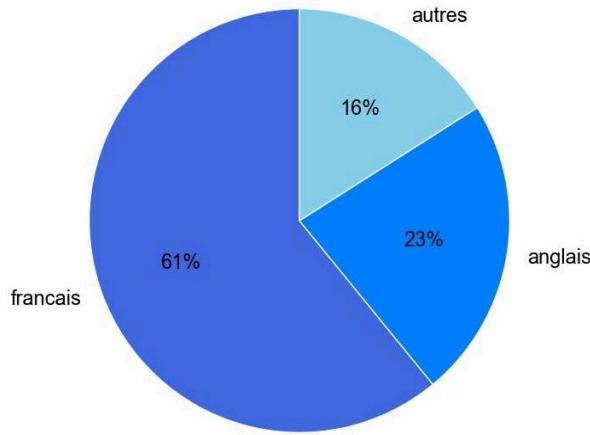
1. Multifonctions LED Aider fer à souder à la main Loupe Pied objectif Loupe outil de serrage Kit de Réparation de soudage et d'assemblage Caractéristiques: Idéal lorsqu'une troisième main est nécessaire. 360 ° cou rotation et les bras flexibles vous permet d'ajuster la lentille à la position où vous en avez besoin maintenant le crocodile plus réglable pinces combinatio support de soudure dans le corps. Une aide utile pour les travaux de soudage ou fabricants modèle. Avec 10 éclairage LED ce modèle peut fonctionner très bien dans l'environnement sombre. compartiment de la batterie inférieure est équipée d'une plaque d'acier qui

- 
- peut améliorer la stabilité. Avec trois pièces de piles AAA grande capacité ce modèle peut se décharger en continu plus de 10 heures opportunément être utilisé. poste de travail parfait pour le soudage les modèles de construction réparation électronique les projets de bricolage passe-temps et de l'artisanat Spécifications: Matériel: Acrylonitrile Butadiène Styrene métal verre optique acrylique. Taille: 205mm x 95mm x 170mm Grossissement: 3 fois (Grande lentille) 45 fois (petit verre) Diamètre de l'objectif: 75 mm (grand) 20 mm (petite) Alimentation par batterie: 3 piles AAA (non incluses) peut également alimenté par câble USB Le forfait comprend: 1 x LED type de clip Aider Loupe à main 3x 45x 1 x câble USB
2. Taille: En format A5 (144 cm x 21 cm) Caractéristique: -Excellent durabilité avec couverture solide design. 96 pages jaune pale (recto et verso) -Texture de papier de haute qualité vous pouvez y écrire facilement
  3. Die Premium-Münzkapseln ULTRA werden aus besonders kratzfestem und glasklarem Acryl hergestellt. Die Kapseln ohne Griffrand sind für alle gängigen Münzen mit einem Durchmesser 36 mm. Festgreifender und gleichzeitig leicht zu öffnender Verschluss. Außendurchmesser 42 mm. Verpackungsinhalt: 10 Stück
  4. MATELAS:  
Accueil : Ferme .  
Soutien : Ferme .  
Technologie matelas : Face été &#43; à face en Mousse Poli

De la même façon que pour les images, les données textuelles peuvent être qualifiées de bruitées. En effet, il n'y a pas de structure commune aux divers textes : ils peuvent être écrits dans différentes langues, comporter des fautes d'orthographe, être exagérément succincts ou à l'inverse excessivement verbeux. Certains ressemblent à des traductions automatiques et ne se focalisent pas toujours sur le produit. Enfin les balises HTML et les caractères spéciaux liés au site en lui-même polluent la lecture : il n'est pas toujours facile, y compris pour un humain, de décrypter certaines descriptions. Nous allons réaliser quelques nouvelles études pour quantifier cela.

### **a) Étude sur les langues au sein du dataframe**

Nous avons utilisé la librairie ‘Langdetect’ qui utilise un algorithme naïf bayésien pour déterminer quelle langue est utilisée dans un texte donné. Nous avons ainsi pu constater la présence de plus de 30 langues. Voici une représentation de leur répartition :

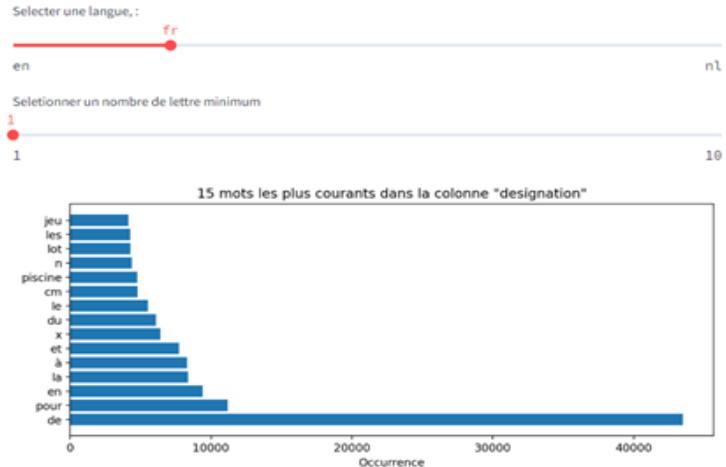


Le français et l'anglais sont les deux langues archi majoritaires et représentent à elles deux 84% du total des données. Il semble opportun de se limiter à elles pour la phase de traitement des données.

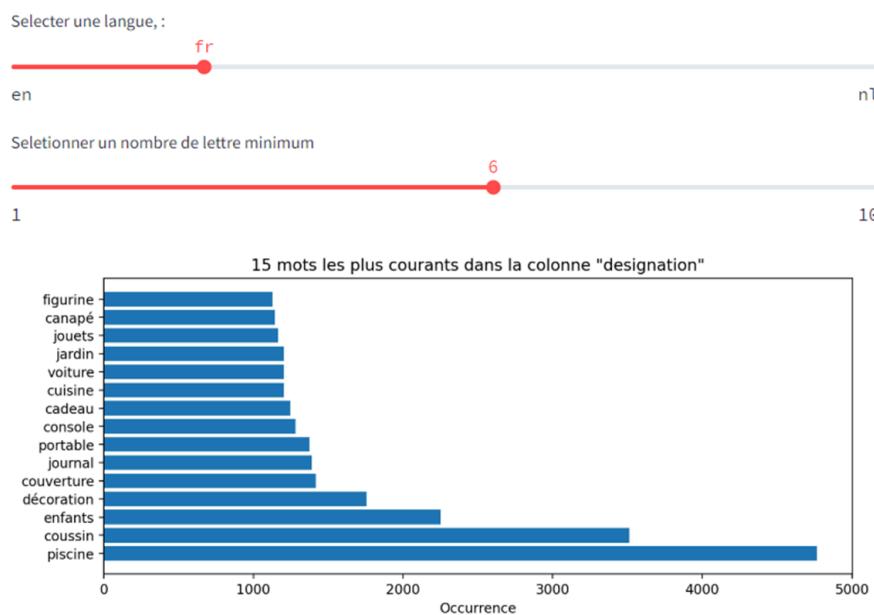
### **b) Répartition de la fréquences des mots dans ‘désignation’**

L'objectif est de capturer naïvement des mots qui puissent être attribués à des labels de code produit. C'est aussi une manière de voir la persistance des mots les plus représentés lorsqu'ils sont en concurrence avec des mots ayant de plus en plus de lettres. Nous avons utilisé la librairie ‘Spellchecker’ pour sélectionner les mots dans leur langue. Ce dictionnaire de langue ne semble pas fonctionner uniformément selon la langue et cautionne un peu l'analyse.

Voici l'extrait d'une présentation ‘Streamlit’ qui permet d'afficher les 15 mots les plus représentés dans la variable explicative ‘désignation’ en fonction de la langue, ici le français:

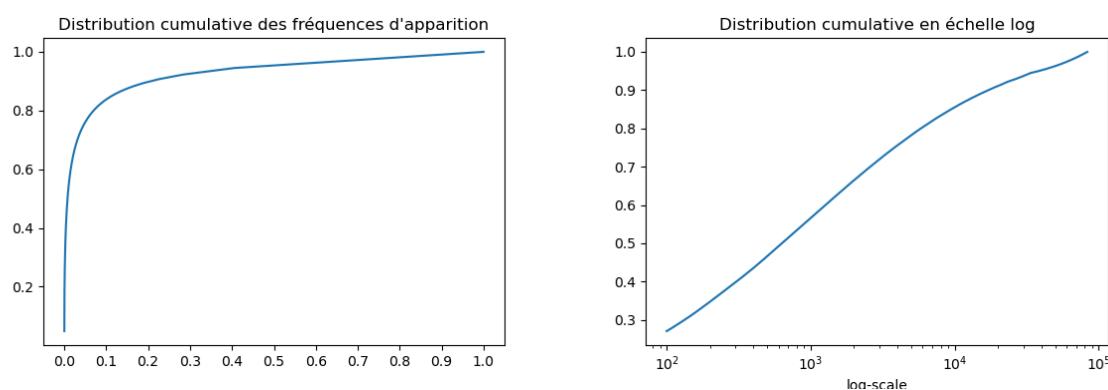


Les mots de plus de 6 lettres commencent à faire apparaître des termes plus propices à une classification:



Si nous prenons l'exemple du mot “piscine”, il est ultra dominant jusqu'à plus de 6 lettres. Au-delà de 9 lettres, les termes désignant des d'objets semblent laisser place à des classes d'objets : décoration, rangement...

Plus généralement, si l'on trie l'ensemble des mots d'un corpus par fréquence d'apparition, on retrouve une relation empirique connue sous le nom de loi de Zipf :  $f \sim K/n$ , où  $f$  désigne la fréquence d'apparition du mot,  $n$  son rang dans l'ordre des fréquences et  $K$  une constante. Par conséquent, la distribution cumulée des fréquences devrait prendre la forme d'une droite en échelle log, ce qui est bien le cas ici :



Nous constatons à nouveau que certains mots sont caractérisés par une forte récurrence, 5% parmi eux suffisent à couvrir 75% du texte des annonces. Ces mots sont en majorité

dépourvus d'intérêt pour notre objectif de classification : il s'agit entre autres de prépositions, d'articles ou de pronoms qui coordonnent les mots dans la langue naturelle mais ne fournissent pas de caractéristiques utiles sur leur sujet.

En fin de support se trouvent à l'inverse une longue liste de mots très rares: fautes d'orthographe, sigles parasites, nouvelle langue... Les mots informatifs se situent donc entre ces deux extrêmes, cela nous incite à établir un critère de pertinence fondé sur l'occurrence des mots en excluant ceux qui sont trop ou trop peu fréquents.

### c) Mots les plus fréquents par classe

Nous décidons de visualiser les mots les plus fréquents pour chaque classe de notre dataframe. Pour se faire, nous utilisons une technique de ‘nuage de mots’ après avoir supprimé les mots sans intérêts (stop\_words) et ceux de moins de 3 lettres:





#### d) Proposition de définition des classes

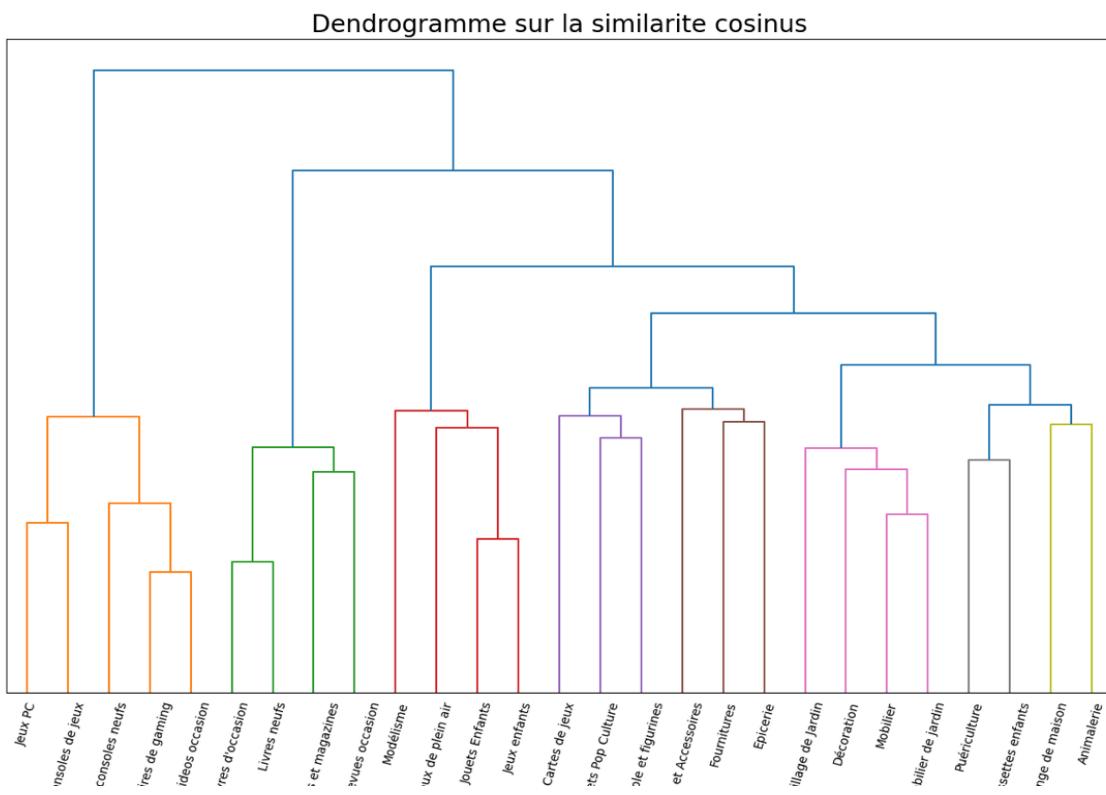
Sur la base de l'exploration du catalogue, nous nous sommes essayés à subsumer la catégorie sous-jacente de chaque code. Voici notre proposition :

<b>Code</b>	<b>Hypothèse</b>	<b>Code</b>	<b>Hypothèse</b>
<b>2583</b>	Brico, Jardin, Animalerie > PISCINE & SPA	<b>1320</b>	Jouet, Enfant, Puériculture > PUERICULTURE
<b>1560</b>	Maison > DECORATION & MOBILIER	<b>10</b>	Livre > FOURNITURE ET MANUEL SCOLAIRE
<b>1300</b>	Jouet, Enfant, Puériculture > UNIVERS MINIATURE	<b>2705</b>	Livre > ROMAN & BD
<b>2060</b>	Maison > DÉCORATION	<b>1140</b>	Jouet, Enfant, Puériculture > FIGURINE & MANGAS
<b>2522</b>	Livre > FOURNITURES PAPETERIE	<b>2582</b>	Maison > JARDIN & EXTÉRIEURS
<b>1280</b>	Jouet, Enfant, Puériculture > OCCASIONS	<b>1302</b>	Jouet, Enfant, Puériculture > JEUX DE PLEIN AIR
<b>2403</b>	Livre > OCCASION	<b>1281</b>	Jouet, Enfant, Puériculture > JEU ÉDUCATIF ET INTERACTIF
<b>2280</b>	Livre > SOCIÉTÉ ET CULTURE	<b>50</b>	Jeux vidéo, Console > ACCESSOIRES
<b>1920</b>	Maison > LINGE DE MAISON	<b>2462</b>	Jeux vidéo, Console > JEUX & CONSOLE
<b>1160</b>	Jeux vidéo, Console > CARTE DE JEUX	<b>2905</b>	Jeux vidéo, Console > SECONDE MAIN
<b>60</b>	Jeux vidéo, Console > RETRO	<b>2220</b>	Brico, Jardin, Animalerie > ANIMALERIE
<b>1180</b>	Jouet, Enfant, Puériculture > DÉGUISEMENT & MAGIE	<b>40</b>	Jeux vidéo, Console > JEUX
<b>2585</b>	Brico, Jardin, Animalerie > BRICOLAGE	<b>1940</b>	Alimentation, Boisson
<b>1301</b>	Jouet, Enfant, Puériculture > JEUX DE CAFE & TEXTILE PUÉRICULTURE		

Une nouvelle fois, il n'est pas évident de tracer la frontière de certaines classes, quand certaines se montrent pléthoriques (la 1301 par exemple: Jouet, Enfant, Puériculture > JEUX DE CAFE & TEXTILE PUÉRICULTURE), d'autres paraîtront redondantes (2403: Livre > OCCASION et 2280: Livre > SOCIÉTÉ ET CULTURE). Pour cette raison, nous nous sommes essayés à un clustering des labels par texte pour étudier l'utilité d'un regroupement des classes.

### e) Clustering textuel des classes

L'approche a consisté à tokeniser seule la colonne 'désignation' regroupée par labels. Pour distinguer au mieux les classes, nous avons vectorisé les tokens selon l'algorithme TF-IDF : Term Frequency, Inverse Document Frequency (voir section preprocessing). Nous avons ensuite réalisé un regroupement hiérarchique en utilisant comme distance la similarité cosinus ce qui nous a permis de tracer le dendrogramme suivant :



La structure qui se dessine confirme nos hypothèses en regroupant des classes qui sont naturellement proches :

Multimédia	Librairie	Jeux	Extérieur
Accessoires	Informatique	Enfants	Intérieur

La faible distance inter-classes pour certains labels confirme la gageure dont relève ce challenge : il faudra mettre en place un modèle de prédiction suffisamment performant pour distinguer des classes qui se confondent facilement.

---

### **3. Conclusion**

L'exploration des données a révélé une richesse et une complexité significatives. Outre la mise en œuvre des outils de visualisation pour caractériser nos données, nous avons ressenti le besoin de mettre en place des approches spécifiques. L'identification des codes produit anonymes par le biais du clustering et de la labellisation, ainsi que l'extraction de ces codes labellisés par l'analyse de la fréquence de mots récurrents et persistants dans plusieurs langues, en utilisant des approches à la fois statiques et dynamiques, nous a paru être une stratégie pertinente pour établir une base relationnelle dans les données et définir des points d'attention spécifiques pour leur modélisation. Naturellement, l'appropriation détaillée des contenus de données, article par article, a été essentielle pour nous permettre d'identifier et de caractériser les éléments parasites, tels que la présence de balises HTML dans le texte ou de contours blancs dans les images.

A ce stade de l'analyse, ces techniques nous ont aidé à identifier les défis majeurs pour la modélisation. Mais face à la gestion du déséquilibre des classes, des hétérogénéités de contenu entre les articles d'une classe et entre classes, des gémellités de classes, du bruits textuels et dans les images, les grandes quantité de données, dans un contexte d'une classification multi-classes, nous avons saisi l'opportunité unique d'appliquer et de développer des techniques avancées de machine learning.

Nous avons débuté par la modélisation textuelle, en appliquant préalablement des techniques de prétraitement.

## **III. Classification unimodale du texte**

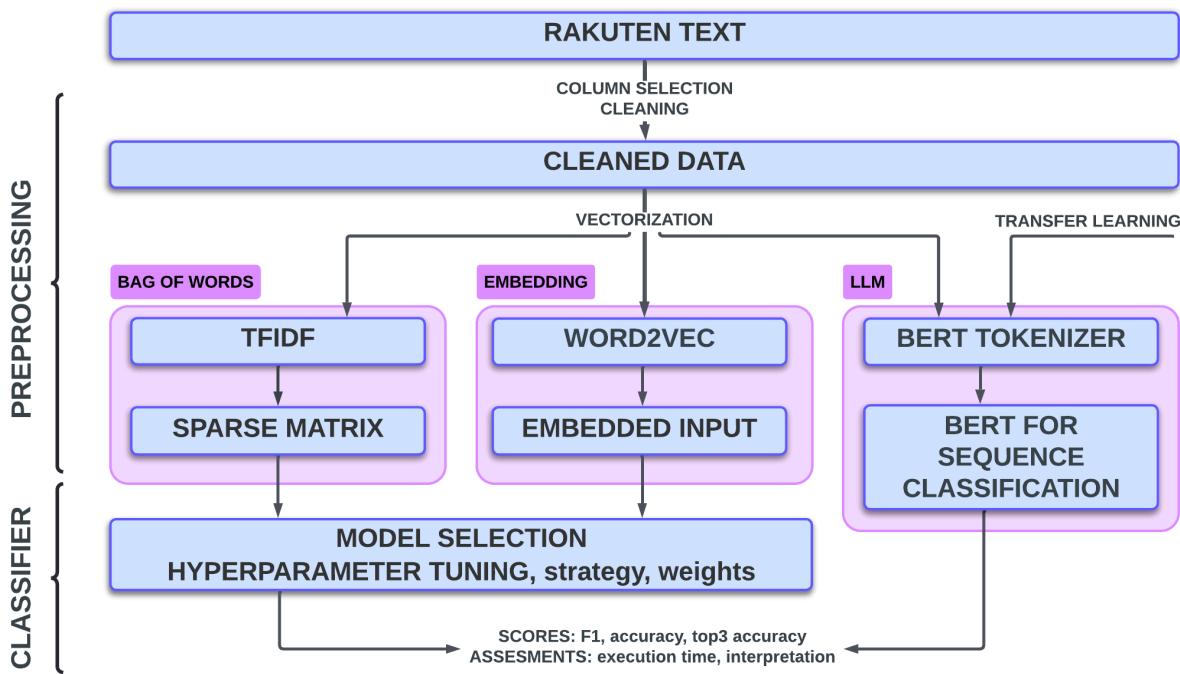
### **1. Pipeline pour la classification texte**

Rappelons que les données texte correspondent dans notre jeu de données à des annonces (une désignation ainsi qu'une description laissée vide dans près de 30% des cas). Ces textes sont écrits pour des utilisateurs humains : nous sommes dans le domaine du traitement du langage naturel (Natural Language Processing, ou NLP).

Ce type de donnée n'est pas directement appréhendé par les modèles standard de machine learning. Il convient pour y parvenir de préparer et transformer le texte brut au cours d'étapes de nettoyage et de tokenisation. Les deux principales méthodes de vectorisation Bag of Words et Word Embedding seront testées et comparées.

La grande dimension des données nous amène ensuite à considérer dans le preprocessing le potentiel bénéfice d'une sélection de caractéristiques postérieure à la vectorisation. Finalement, nous étudierons plusieurs stratégies pour adapter les modèles standards à notre cadre qui est celui d'une classification avec un très grand nombre de classes déséquilibrées entre elles.

Nous résumons dans la figure suivante le cheminement de ces diverses étapes :



## 2. Nettoyage et tokenization du texte

Le processus de tokenisation consiste à délimiter et découper le texte en unités plus petites qui seront ensuite numérisées. La méthode la plus intuitive est de décomposer en mots, mais il existe d'autres façons de faire.

Par exemple, le bigramme "hot dog" n'a pas le même sens que l'adjonction de "hot" et de "dog". Cette observation nous incite à considérer l'éventuelle utilité de séquences de n mots, appelées n-grammes.

---

Cette étape s'accompagne d'un nettoyage du texte qui vise à le débarrasser du bruit qu'il peut contenir. En effet, la phase d'exploration nous a appris que ces annonces sont émaillées de variations subtiles qui font la richesse de la langue écrite mais risquent aussi de faire saturer nos modèles.

Ainsi, nous pouvons dessiner le contour d'une représentation stable du texte et sélectionner uniquement les tokens potentiellement utiles. Nous revenons ci-dessous sur plusieurs axes étudiés :

❖ **retrait des espaces excessifs et mise en minuscules**

On neutralise les espaces en trop et on transforme toutes les majuscules en minuscules. Sans cette opération, un même mot pourrait faire l'objet d'une tokenisation différente selon qu'il se trouve en début ou milieu de phrase.

<b>DATA</b>	1 Housse De Couette 220x240 Cm &#43; 2 Taies D&#39;Oreillers 63x63 Cm
<b>PROCESSING</b>	1 housse de couette 220x240 cm &#43; 2 taies d&#39;oreillers 63x63 cm

L'utilité de cette première étape ne fait pas débat, bien qu'elle puisse déjà retirer des nuances utiles, comme la différence entre "Mars" (la planète) et "mars" (le mois).

❖ **balises et caractères HTML**

Au vu de l'importante présence de balises HTML, nous avons émis l'hypothèse que de nombreux textes étaient simplement recopiés du code web des annonces. Il est naturel de considérer parasitaire cette syntaxe qui est d'ailleurs invisible sur l'interface utilisateur.

Nous neutralisons ce code avec le parser HTML de BeautifulSoup :

<b>DATA</b>	1 housse de couette 220x240 cm &#43; 2 taies d&#39;oreillers 63x63 cm
<b>PROCESSING</b>	1 housse de couette 220x240 cm + 2 taies d'oreillers 63x63 cm

Remarquons que le code HTML n'est pas totalement polluant du point de vue d'un classifieur. Au contraire, il signale essentiellement des annonces publiées par des professionnels, ce qui délimite justement certaines catégories. Néanmoins il n'est pas souhaitable que notre modèle apprenne de ce type de biais propres au dataset Rakuten.

---

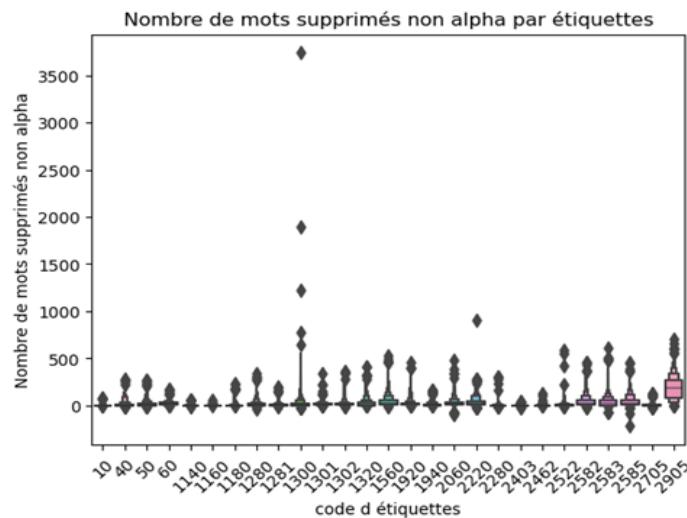
#### ❖ retrait des nombres et des caractères spéciaux

Les valeurs numériques ne comportent en général pas d'informations de premier plan pour la classification : dimensionnement, date, quantité d'objets, code HTML...

Nous ne gardons que les mots sans chiffres qui peuvent contenir les signes diacritiques de langue française ou anglaise.

<b>DATA</b>	1 housse de couette 220x240 cm + 2 taies d'oreillers 63x63 cm
<b>PROCESSING</b>	housse de couette x cm taies d oreillers x cm

Ce filtre possède l'inconvénient de supprimer les dates. Les marqueurs temporels concernent en effet de façon inégale les étiquettes, comme par exemple les titres de livre qui possèdent davantage de termes numériques ou de mots non alpha par étiquette.



#### ❖ filtrage des mots-vides

Pour rappel, les mots vides sont les mots courants dont nous avons compris la faible valeur informative. En les ignorant nous allégeons considérablement la tokenisation :

<b>DATA</b>	housse de couette x cm taies d oreillers x cm
<b>PROCESSING</b>	housse couette taies oreillers

Les mots de moins de deux ou trois lettres sont aussi généralement vus comme des mots vides. Nous nous bornerons aux stop-words usuels des langues française et anglaise.

---

#### ❖ langues étrangères, racinisation et lemmatisation

A ce stade, nous pouvons encore améliorer en normalisant les tokens par racinisation ou lemmatisation. Ce procédé qui consiste à réduire les mots à leur racine linguistique ou lemme permettrait dans l'exemple ci-dessus de neutraliser le pluriel des deux derniers mots :

<b>DATA</b>	housse couette taies oreillers
<b>PROCESSING</b>	houss couet tai oreiller

Notons que le processus de normalisation diffère selon la langue considérée, il va donc falloir nous adapter selon que le texte est en anglais ou en français. Les autres langues sont trop rares et nous choisissons de les ignorer.

### **3. Vectorisation par sac de mots**

#### **A. Principe**

La représentation en sac de mots consiste tout simplement à vectoriser les tokens en comptant leur nombre d'occurrences ou leur fréquence. En guise d'exemple prenons deux désignations issues du jeu de données:

##### **DATA ORIGINALE :**

$X_{2183}$  : "Je Vends Ma Playsation 3 Slim + 1 Manette + 7 Jeux " (sic)

$X_{23\ 354}$ : "Vends Lot De 7 Tomes Sur Les Sciences Occultes"

##### **DATA APRÈS PREPROCESS :**

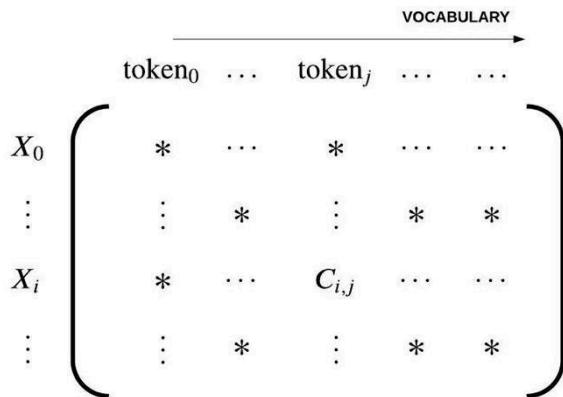
$X_{2183}$  : [vends, playsation, slim, manette , jeux]

$X_{23\ 354}$  : [vends, lot, tome, science, occultes]

##### **DATA APRÈS VECTORISATION :**

DATA	JEUX	LOT	MANETTE	OCCULTES	PLAYSATION	SCIENCE	SLIM	TOME	VENDS
2183	1		1		1		1		1
23354		1		1		1		1	1

Plus généralement cela conduit à transformer les données en une matrice creuse  $C$ , dans la mesure où chaque annonce ne contient en fait qu'une fraction de tout le vocabulaire possible :



avec le coefficient  $C_{i,j}$  non nul si et seulement si le  $\text{token}_j$  apparaît dans l'annonce  $X_i$ .

La vectorisation TF-IDF (pour Term Frequency – Inverse Document Frequency) est un raffinement cherchant à introduire une notion de pertinence. Les coefficients de la matrice sont pondérés par l'inverse de la fréquence d'apparition du token dans le reste du corpus :

$$C_{i,j} = TF_{i,j} * IDF_j$$

$TF_{i,j}$  : Term-Frequency, fréquence d'apparition du  $\text{token}_j$  dans  $X_i$

$IDF_j$  : Inverse-Document-Frequency, un facteur inversement proportionnel à la fréquence d'apparition du  $\text{token}_j$  dans l'ensemble du corpus (plusieurs méthodes de calcul sont possibles)

C'est le même principe que pour le filtrage stop-words : on cherche à diminuer l'influence des mots pas suffisamment spécifiques. A l'opposé, les termes rares au sein du corpus (à DF faible, donc IDF élevé) sont revalorisés. Dans le précédent exemple, les poids des coefficients de la colonne 'vends' auraient ainsi été réduits.

## B. Implémentation avec Scikit-learn

Les classes CountVectorizer et TfidfVectorizer du sous module 'feature\_extraction.text' de scikit-learn permettent de réaliser la vectorisation de manière très pratique et peuvent aussi se charger d'une part du prétraitement depuis l'appel des paramètres l'instance :

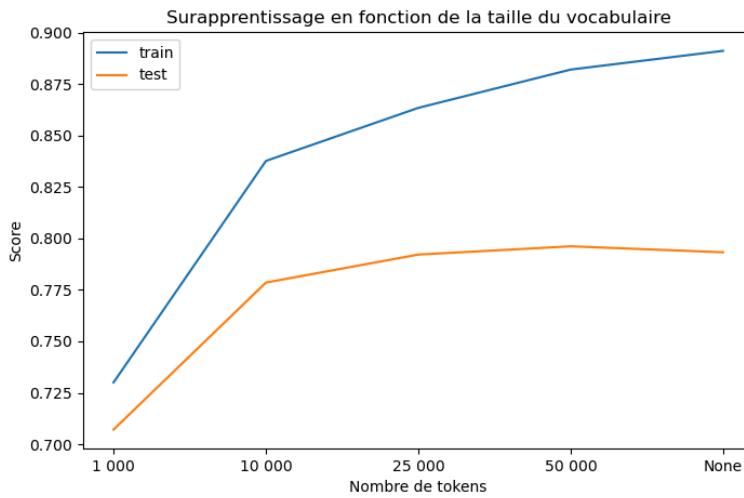
---

<b>Paramètre</b>	<b>Valeur par défaut</b>	<b>Description</b>
ngram_range	(1, 1)	plage des n-grammes
token_pattern	\b\w\w+\b	expression régulière définissant les tokens ici : mots d'au moins 2 caractères
stop_words	None	liste custom de mots à supprimer

En outre, les paramètres ci-dessous permettent de réaliser une sélection des caractéristiques sur le principe des statistiques d'apparition des tokens :

<b>Paramètre</b>	<b>Valeur par défaut</b>	<b>Description</b>
max_df	1.0	seuil maximal pour la fréquence d'apparition
min_df	1	nombre d'occurrences minimal
max_features	None	limite le nombre total de termes

En effet, moins nous filtrons de mots, plus le vocabulaire considéré sera important, et plus notre représentation sera sujette au fléau de la dimension. Pour illustrer cela, nous considérons dans la figure suivante les performances d'une régression logistique sur notre jeu de données avec pour simple prétraitement la vectorisation par défaut de la classe `tfidfVectorizer` en faisant varier le paramètre `max_features` :



Cette figure illustre bien que l'erreur de généralisation croît avec la taille du vocabulaire, ce qui est prévisible : le modèle se sur-ajuste sur tous les mots “parasites”.

Nous examinerons de surcroît la réduction de dimension de la matrice post hoc par des techniques de sélection de features, comme l'analyse de l'information mutuelle avec SelectKBest. Nos divers essais nous ont montré que la réduction de dimension mathématique peut s'avérer utile mais ne suffit pas à elle seule : une réduction préalable de dimension par nettoyage linguistique reste nécessaire pour garantir des performances correctes.

### C. Conclusion de la phase de preprocessing

Le prétraitement est ici à la fois délicat et critique : il faut trouver le juste compromis en éliminant suffisamment de bruit tout en conservant les informations importantes. Nous avons statué sur le choix d'une approche “performance-driven” en ne réalisant que les étapes absolument nécessaires pour éviter la saturation du vocabulaire et un sur-apprentissage, comme la mise en minuscule ou le retrait du HTML, des caractères spéciaux et des stop words usuels. L'utilisation d'autres traitements a été décidée à partir de critères de simplicité et d'amélioration de performances :

- ❖ choix de la colonne : prendre l'ensemble ‘designation’ + ‘description’ aide dans de nombreux cas mais peut rajouter du bruit dans d’autres ce qui en moyenne se compense. Cela complexifie dans tous les cas la pipeline, nous décidons en conséquence de ne considérer que l'information synthétique contenue dans désignation

- 
- ❖ normalisation de texte : la gestion des langues ainsi que les techniques de normalisation de texte n'améliorent pas significativement les performances
  - ❖ vectorisation : la méthode TF-IDF couplée d'une limitation du vocabulaire réduit plus efficacement le surapprentissage que le sac de mots basique

## D. Modélisations expérimentées

Maintenant que les principes généraux de prétraitement ont été évoqués, voici les différents algorithmes que nous avons expérimentés :

MODÈLE	HYPERPARAMÈTRES
K-Neighbors	<b>n_neighbors</b> : [3, 5, 8] <b>metric</b> : ['euclidean', 'manhattan', 'minkowski']
Logistic Regression	<b>penalty</b> : ['l1'] ou <b>penalty</b> : [ 'l2',] <b>C</b> : [0.1, 1] <b>solver</b> : ['liblinear'] ou <b>solver</b> : ['saga']
Random Forest	<b>n_estimators</b> : [300, 1000] <b>max_features</b> : ['sqrt', 'log'] <b>min_samples_leaf</b> : [1, 3, 5]
XGBoost	<b>booster</b> : 'gbtree'
Neural Network	<b>Dense layers</b> : 4, <b>Dropout layers</b> : 2 <b>learning_rate</b> : 1e-3
Linear SVC	<b>penalty</b> : ['l1'] ou <b>penalty</b> : [ 'l2',] <b>C</b> : [0.1, 1]
Naive Bayes	<b>alpha</b> : [0.01, 0.1, 1]

Nous avons ajouté à cette liste de modèles classiques deux classificateurs un peu moins répandus mais réputés pour afficher de bonnes performances en classification de texte :

- 
- ❖ LinearSVC :

---

Cet algorithme est une variation des SVM classiques. Il se concentre sur les séparations linéaires plutôt que d'utiliser une fonction de noyau, ce qui s'avère plus efficace pour les grands ensembles de données avec de nombreuses caractéristiques.

❖ **Naïve Bayes** :

De façon très synthétique, le classifieur estime pendant l'entraînement les probabilités a priori d'appartenir à chaque classe  $P(C_i)$  et les probabilités pour chaque token d'appartenir à ces classes  $P(token | C_i)$ . Puis en s'inspirant du théorème de Bayes, il calcule un score :

$$P(C_i | \text{annonce}) \sim \prod P(token | C_i)$$

Cette formule justifie que la méthode soit dite “naïve”, dans la mesure où elle ne devrait être valide que si les tokens étaient indépendants, ce qui n'est pas le cas.

Enfin, voici un exposé des méthodes testées (lorsque cela est possible) pour tenir compte du cadre particulier de notre classification caractérisée par un grand nombre de labels déséquilibrés :

MÉTHODE	PRINCIPE
Rééchantillonnage	rééquilibrer les tailles des classes, soit en supprimant des échantillons de celles qui sont majoritaires, soit en générant pour celles qui à l'inverse sont minoritaires
Bagging et Boosting	utiliser des algorithmes d'ensemble qui peuvent améliorer la robustesse et la précision en combinant les prévisions de plusieurs modèles de base

---

Pondération des Classes	modifier la fonction de perte pour pénaliser plus sévèrement les erreurs sur les classes minoritaires
Stratégie Multi-Classes	entraîner des classificateurs distincts pour chaque paire de classes (One-vs-One) ou pour chaque classe contre toutes les autres (One-vs-Rest)

## E. Résultats

Nous comparons les comportements généraux de nos modèles en étudiant les résultats moyens obtenus avec ces diverses combinaisons d'hyperparamètres:

Modèle	Train Accuracy	Validation Accuracy	Train Weight F1	Validation Weight F1	Temps
K-Neighbors	69,5 % ± 3,7%	59,5 % ± 4,1%	71,2 % ± 3,8%	61,3 % ± 3,6 %	5 s
Logistic Regression	70,0 % ± 7,0%	67,7 % ± 5,6%	71,2 % ± 3,8%	61,3 % ± 3,6 %	5 s
Random Forest	71,1 %	70,9 %	71,7 %	71,5 %	4ms
XGBoost	82,9 %	72,9 %	83,7 %	73,7 %	1,5 min
Neural Network	80,3 %	75,6 %	80,4 %	75,5 %	3 min
Linear SVC	71,7 % ± 2,1%	70,0 % ± 1,7%	71,3 % ± 2,4%	69,6 % ± 2,0 %	1 s
Naive Bayes	67,8 % ± 2,8%	66,0 % ± 2,2%	65,2 % ± 3,1%	63,3 % ± 2,4 %	0,1 s

- ❖ Plafonnement des scores autour de 70-75%, avec K-Neighbors un peu en dessous. Le niveau général de performances est bon compte tenu du fait qu'il y a 27 classes

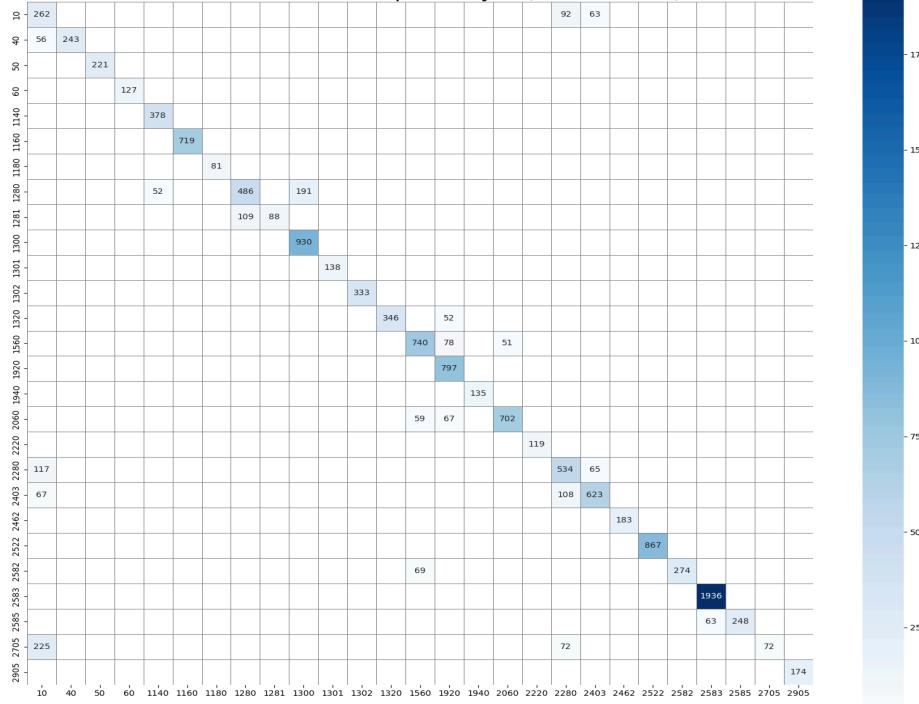
- 
- ❖ Les écart type en validation croisée sont faibles, suggérant que le volume des données est suffisant pour que les algorithmes optimisent leurs capacités
  - ❖ K-Neighbors, Logistic Regression et les réseaux de neurones sont plus prônes au sur-apprentissage sans sélection de caractéristiques. Comme attendu, les méthodes avec forte régularisation L1 ont moins besoin de recourir à cette méthode
  - ❖ Les réseaux de neurones sont plus longs à entraîner. Les méthodes bayésiennes se distinguent par une grande efficacité temps au regard de leur simplicité et de leur rapidité

En recherchant la combinaison optimale d'hyperparamètres pour trois de ces modèles, nous avons obtenu les résultats suivants :

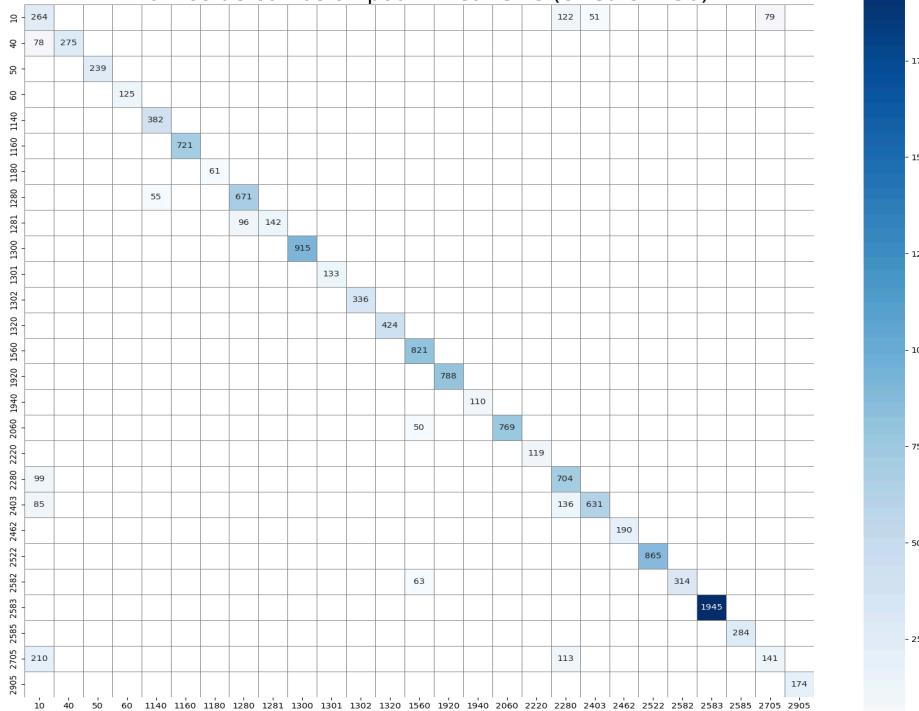
- ❖ **Random Forest** : random undersampling, n\_estimators : 1000
- ❖ **Complement Bayes**: alpha:1, au préalable SelectKBest k : 3000
- ❖ **LinearSVC** : régularisation L1, C : 0.1

Modèle	Weight Accuracy	Macro Accuracy	Top 3 Accuracy	Weight F1	Macro F1
Random forest	71.1 %	70.9 %	85,2 %	71.7 %	71,5%
Bayes	69,2 %	66,3 %	84,6 %	67,9 %	65,0 %
Linear SVC	73,9 %	69,5 %	85,7 %	73,7 %	70,8 %

## Matrice de confusion pour Bayes (erreurs > 50)



Matrice de confusion pour Linear SVC (erreurs > 50)



---

Le classifieur Bayésien, en plus d'être rapide, produit des scores proches des meilleurs. Il retient notre attention dans le cadre d'un déploiement léger et efficace, alors que dans l'optique de la performance pure les deux autres modèles conservent l'avantage. En particulier, Random Forest se comporte mieux au niveau du F1, ce qui traduit une meilleure gestion classe par classe.

Comme nous nous y attendions, tous les modèles concentrent leurs erreurs de classification sur des confusions entre labels proches du point de vue des champs lexicaux, comme 10 (livres et fournitures), 2705 (romans) et 2280 (livres société), ou alors 1280 (Jouet > OCCASIONS) et 1281: (Jouet > JEU ÉDUCATIF ET INTERACTIF). Nous observons également que la qualité du texte impacte directement le taux de bonnes prédictions. Les annonces dont le texte n'est pas suffisamment fourni ou précis sont nettement moins bien gérées dans l'ensemble.

Finalement, pour des modèles simples, dans le sens où ils sont fondés sur une représentation simplement fréquentiste des mots, les résultats sont tout à fait honorables tant en performance qu'en rapidité du calcul au regard de la complexité de la tâche (traitement du langage naturel et très nombreuses classes).

## **F. Interprétation avec Lime**

### **Interprétabilité avec le package lime pour les résultats logistic regression**

L'interprétabilité ci-dessous prend les 10 features (mots) identifiés comme les plus pertinents pour réaliser la prédiction pour chacune des tops 2 classes prédites.

### **Contraintes pour réaliser une classification automatisée**

La difficulté de notre exercice réside dans le fait que les désignations et descriptions ne sont pas rédigées avec une intention de classification. Ces champs sont libres pour permettre à l'internaute d'être le plus pertinent par rapport au produit qu'il souhaite vendre. Cette liberté est importante pour ne pas brider l'internaute qui est celui qui connaît le mieux son produit, cependant, cela peut être une problématique dans notre exercice car l'internaute ne connaît pas nécessairement les classes définies par Rakuten.

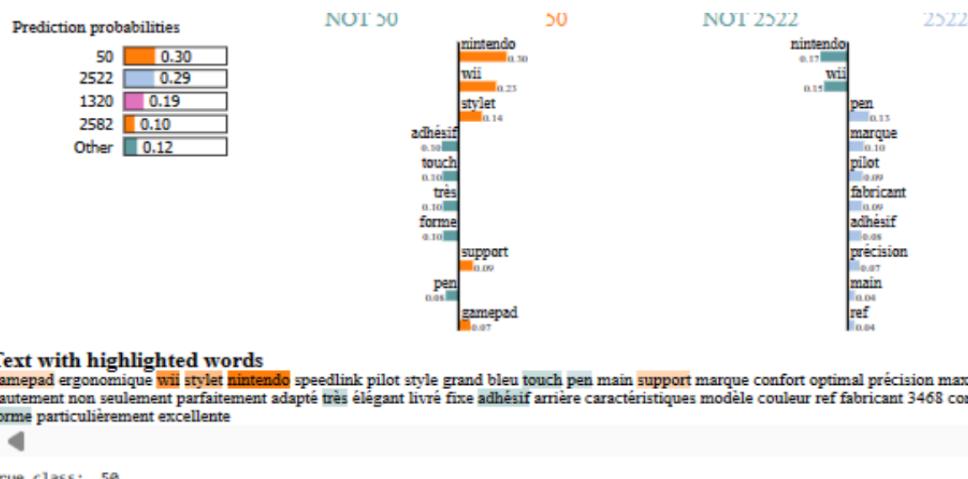
### **Cas particulier de notre projet**

Les classes définies dans notre projet peuvent présenter une forte adhérence et entraîner des difficultés pour la classification de nos modèles. Nous avons notamment les classes : 40 (que nous avons défini comme "Jeux video neufs") et 2462 (que nous avons défini comme "Jeux video

d'occasion").

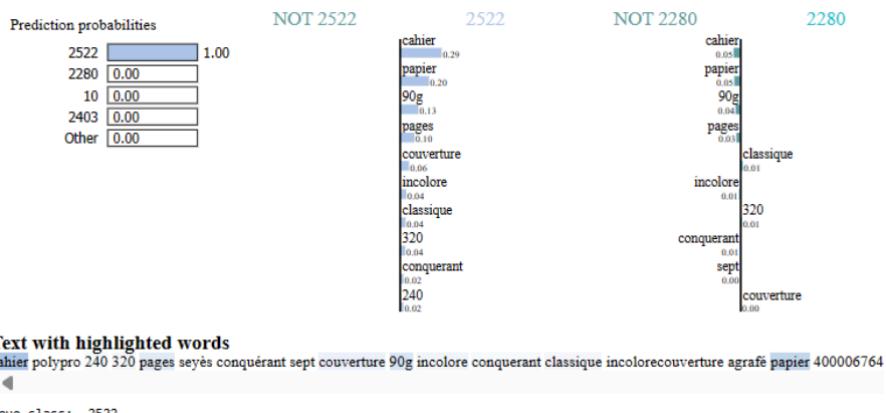
### Exemple 1: Stylet Nintendo

Prédiction faible liée aux termes utilisés qui sont interprétés comme pouvant appartenir à plusieurs classes. La classe prédite est la classe 50 (que nous avons définie comme "Jeux vidéo, Console > ACCESSOIRES") avec une prédition de 0.30. Les termes liés aux jeux-vidéos sont mis en avant par lime, avec "nintendo", "wii", "stylet".



La seconde classe obtenant le meilleur score est la classe 2522 (que nous avons définie comme "Livre > FOURNITURE PAPETERIE") avec une prédition de 0.29 avec une sémantique liée aux fournitures. Les termes amenant à cette classification sont "pen", "marque", "pilot".

### EXEMPLE 2: Cahier



---

Le modèle réalise une prédiction correcte à 100% avec la description saisie. Les termes sont tous identifiés comme appartenant à la bonne classe 2522 (que nous avons définie comme “Livre > FOURNITURE PAPETERIE”).

En conclusion, l’interprétabilité nous rappelle aussi les limites de la représentation en sac de mots : elle n’est pas capable de tenir compte de la grammaire, du sens, du contexte ainsi que de l’importance du prétraitement par rapport à la sémantique qui sera analysée pour définir ses prédictions.

## **4. Vectorisation par plongement lexical**

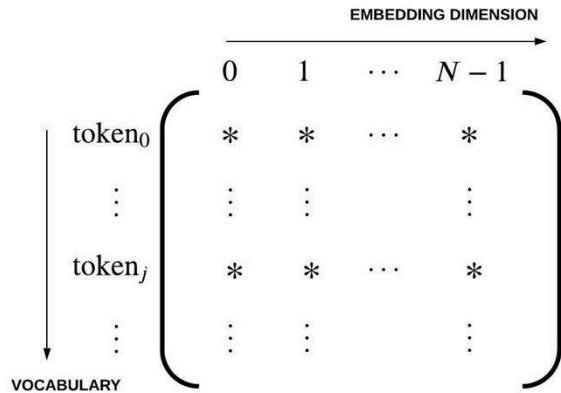
### **A. Principe**

Les limites relevées à l’approche précédente nous amènent à essayer un nouveau type de vectorisation du vocabulaire qui soit en mesure de capturer la sémantique. Intuitivement, on pourrait imaginer un espace vectoriel à N dimensions, dans lequel chaque dimension jouerait le rôle d’une feature porteuse de sens.

Le plongement lexical (word embedding) est la solution apportée à ce défi. Il est fondé sur l’hypothèse selon laquelle des mots apparaissant dans des contextes similaires doivent avoir des significations apparentées. La méthode word2vec est une technique populaire de plongement lexical qui repose sur l’entraînement d’un réseau neuronal. Elle possède deux variantes :

- ❖ **Continuous Bag Of Words** : le réseau neuronal est entraîné pour prédire les mots en fonction de leur contexte
- ❖ **Skip2Gram** : le réseau a pour but de prédire le contexte en fonction du mot central.  
Cette variante bien que moins intuitive est plus rapide à entraîner et plus répandue

Dans les deux cas, la vectorisation souhaitée est obtenue en extrayant la couche cachée du réseau. Finalement, le vocabulaire peut être condensé dans une matrice d’embedding  $E$ :

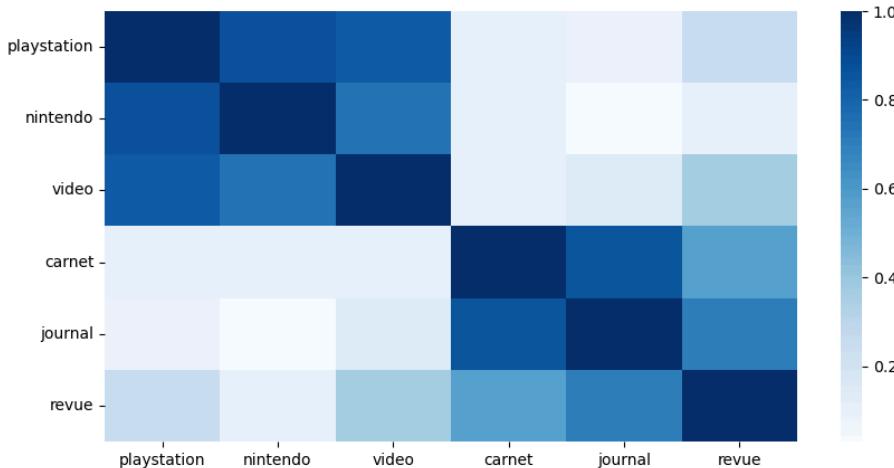


La vectorisation du jeu de données se déduit aisément comme le produit matriciel  $C \times E$ , où  $C$  est la matrice creuse définie dans Bag Of Words. Il est important de noter que la dimension a été drastiquement réduite : elle passe de taille du vocabulaire ( $\sim 10^5$ ) à la taille de l'embedding ( $\sim 50$ ).

## B. Implémentation avec gensim

Le package **gensim** dispose du module **models.Word2Vec** permettant de générer notre propre embedding sur le jeu de données. Le module dispose également d'une fonction de prétraitement sommaire que nous utiliserons en conjonction de celui déjà réalisé plus haut.

À ce stade, nous pouvons vérifier au travers d'une matrice de cosinus similarité sur un échantillon d'embedding que la représentation est bien conforme aux attentes énoncées :



---

Les coefficients observés ci-dessus confirment que des mots qui ont un sens proche sont bien représentés par des vecteurs relativement similaires.

### **C. Résultats et interprétation**

Nous présentons ci-dessous une synthèse de quelques résultats obtenus avec par des modèles utilisés au-dessus de l'embedding :

Modèle	Accuracy	Top 3 Accuracy	Weight F1
SVC	65 %	84 %	63 %
Logistic Regression	62 %	81 %	61%
Neural Network	65 %	84 %	64 %

Ces résultats sont bons mais inférieurs à ceux obtenus par des méthodes fréquentistes, c'est donc une déception de constater que la prise en compte de la sémantique n'a pas apporté d'améliorations. Néanmoins, il nous faut observer que l'embedding est généré ici directement depuis l'ensemble d'entraînement : nous aurions pu opter pour une vectorisation plus sophistiquée en exploitant un modèle déjà entraîné via une approche par transfer learning.

De plus, la notion de contexte a été introduite pour définir un sens, mais les modèles employés continuent de ne traiter le sens que de manière isolée pour chaque mot, mot à mot. Plus concrètement dans l'exemple ; "je vends un livre sur les piscines", l'importance relative de chaque mot n'est pas complètement prise en compte. En résumé, nous nous tournons vers une approche mieux en mesure de saisir le contexte.

## **5. Large Language Model**

### **A. Principe**

Jusque récemment, les réseaux de neurones récurrents étaient les modèles de pointe pour saisir le contexte d'une phrase. Les mots y sont traités séquentiellement et les relations qui les lient peuvent être retenues au travers de mécanismes de mémoire. Toutefois, malgré de bons

---

résultats, les couches récurrentes ont tendance à perdre la dépendance qui peut exister entre des mots quand ils sont trop éloignés, quand bien même ils seraient sémantiquement très reliés.

Dans le sillage du révolutionnaire article “Attention is All you need”, l’architecture Transformer a été introduite pour pallier cet écueil. De façon très synthétique, le Transformer est un réseau qui n’a plus besoin de couches récurrentes : le mécanisme d’attention seul lui permet de tenir compte des dépendances long terme avec succès.

Nous optons pour une démarche de transfer learning utilisant des Transformers déjà entraînés sur de gigantesques volumes de données texte: les Large Language Models (ou LLM). Cela nous permettra de bénéficier des connaissances avancées que ces modèles complexes ont acquises. Le modèle BERT (Bidirectional Encoder Representations from Transformers) développé par Google et ses équivalents français CamemBERT et FlauBERT sont des LLM très utilisés en classification de texte. Nous décidons plus particulièrement d’utiliser une extension dédiée à cette problématique: BertForSequenceClassification.

## **B. Architecture et dimension du modèle**

- ❖ **Intégrations de mots** : le modèle a une taille d'intégration de 768 dimensions et une taille de vocabulaire de 32 000 tokens
- ❖ **Intégrations de positions** : le modèle peut gérer des séquences d'entrée d'une longueur maximale de 512 tokens
- ❖ **Couches d'encodeur** : le modèle comporte 12 couches d'encodeur, chacune avec une taille de couche cachée de 768 et une taille de couche intermédiaire de 3072.
- ❖ **Couche de classification spécifique à une tâche** : le modèle BertForSequenceClassification est conçu pour les tâches de classification de séquence. Il prend l'état caché final du jeton [CLS] et le fait passer à travers une couche linéaire et une fonction softmax pour produire des probabilités de classe. Dans notre cas, le modèle est configuré avec un nombre personnalisé d'étiquettes (Num\_labels= 27) et un vocabulaire spécifique à notre périmètre.
- ❖ **Couches d'encodeur** : La séquence d'entrée intégrée est ensuite traitée via les couches d'encodeur du modèle BERT, qui se composent de mécanismes d'auto-attention et de réseaux neuronaux à action directe. Au cours de ce processus, le modèle apprend à capturer les

---

informations sémantiques et syntaxiques présentes dans la séquence d'entrée, ainsi que toutes les relations entre les tokense.

- ❖ **État caché final de [CLS]** : À la fin des couches d'encodeur du modèle BERT, chaque jeton a un vecteur d'état caché correspondant. Ce vecteur est ensuite transmis à la couche de classification spécifique à la tâche.
- ❖ **Couche linéaire** : L'état caché final du jeton [CLS] est introduit dans une couche linéaire, qui mappe le vecteur à 768 dimensions (en supposant le modèle BERT de base) à un vecteur de taille égale au nombre de classes cibles. Il s'agit essentiellement d'une multiplication matricielle de poids suivie d'une addition de termes de biais.
- ❖ **Fonction Softmax** : La sortie de la couche linéaire passe ensuite par une fonction softmax, qui convertit les valeurs de sortie brutes en probabilités de classe. La fonction softmax garantit que la somme des probabilités dans toutes les classes est égale à 1.

### C. Implémentation avec BertForSequenceClassification

Nous utilisons une fonction de preprocessing intégrée au modèle: ‘dbmdz/bert-medium-historic-multilingual-cased’. Cette dernière, en outre du prétraitement classique (élimination des majuscules et accents, unicode, conversion des nombres en chaînes de texte) réalise une tokenisation spécifique au transformer BERT, permettant de tirer optimalement parti de ses forces. Nous en évoquons ci-dessous quelques aspects importants :

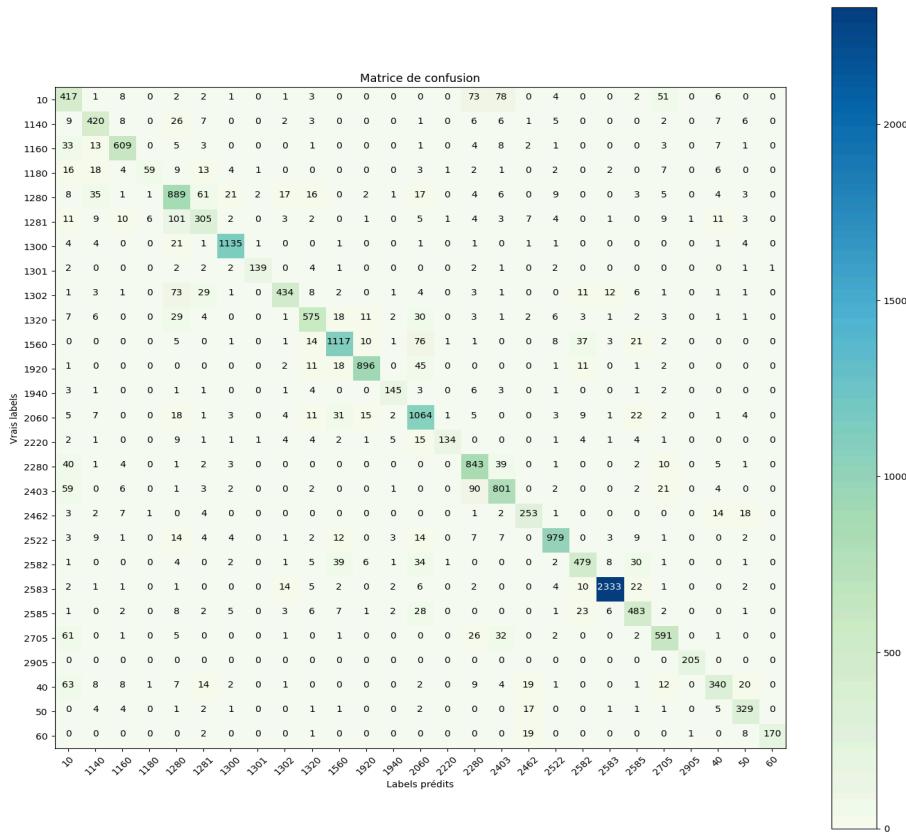
- ❖ **Segmentation de mot en sous-mots** : Les tokens correspondent généralement à des fragments des mots d'usage. Cette granularité offre une plus grande versatilité au modèle (par exemple dans le cas des mots hors vocabulaire) améliorant ainsi ses performances
- ❖ **Utilisation de tokens additionnels** : des tokens spéciaux marquent des informations utiles telles que les débuts et fins de phrase, les segments conditionnels ou les balises de classe. Ce procédé facilite entre autres l'apprentissage de la grammaire. Par exemple, la fonction de prétraitement insère le token [CLS] en début de séquence, et le token [SEP] en fin de phrase.
- ❖ **Longueurs maximales** : des restrictions sur la longueur maximale des séquences sont ajoutées, ce qui permet de maîtriser la complexité computationnelle et d'uniformiser les entrées. Ici le modèle peut gérer des séquences d'entrée d'une longueur maximale de 512 tokens.

## D. Résultats et interprétation

### ❖ Accuracy et F1 score

	<b>eval_loss</b>	<b>eval_Accuracy</b>	<b>eval_F1</b>	<b>eval_Precision</b>
train	0.33	90.3 %	88 %	90 %
val	0.74	79.1 %	76.9 %	79.5 %
test	0.75	78.7 %	75.9 %	78.4 %

### ❖ Matrice de confusion



Nous visualisons un taux d'erreur de prédition relativement faible même si l'algorithme paraît en difficulté sur des labels très proches tels le 1280 : "Jouet, Enfant, Puériculture > OCCASIONS" et le 1281: "Jouet, Enfant, Puériculture > JEU ÉDUCATIF ET INTERACTIF". Nous observons de bons résultats, mais pas significativement meilleurs au vu de la complexité ajoutée. En effet, la progression en performances s'est faite au prix d'une perte de compréhension : l'espace appris n'est plus directement interprétable.

---

## **6. Conclusion**

La classification unimodale du texte a été explorée en profondeur, en commençant par un prétraitement de la donnée afin de sélectionner les contenus pertinents pour la modélisation. Les données textuelles ont donc été nettoyées puis tokenisées et vectorisées en utilisant deux méthodes principales : le sac de mots et le plongement lexical en ajustant le nombre de tokens au performance des modèles.

Nous avons expérimenté une variété de modèles de classification, y compris K-Neighbors, Logistic Regression, Random Forest, XGBoost, Neural Network, Linear SVC et Naive Bayes, avec différents hyperparamètres. Nous avons également exploré des méthodes pour gérer le déséquilibre des classes, comme le rééchantillonnage, le bagging et le boosting, la pondération des classes et les stratégies multi-classes. Les résultats ont montré que les performances générales des modèles étaient bonnes, avec des scores d'accuracy et de F1 pondérés atteignant environ 70-75%, en optimisant la combinaison d'hyperparamètres pour certains modèles.

L'interprétabilité avec Lime nous a été utile car elle met en évidence les limites de la représentation en sac de mots, qui ne tient pas compte de la grammaire, du sens, du contexte et de l'importance du prétraitement pour l'analyse sémantique.

Enfin, l'approche LLM en utilisant BertForSequenceClassification fournit de bons résultats, elle met également en évidence le compromis entre la performance et l'interprétabilité sémantique dans les modèles d'apprentissage automatique.

Nous avons continué le défi par la modélisation des images, en appliquant préalablement des techniques de prétraitement.

---

# **IV. Classification unimodale des images**

## **1. Généralités**

Les images sont des données complexes composées de pixels organisés en matrices tridimensionnelles (largeur, hauteur, couleurs). Avant de les utiliser pour entraîner un modèle, il est crucial de les prétraiter convenablement afin d'extraire les caractéristiques pertinentes et de standardiser leur format.

Le deep learning a révolutionné le traitement des images grâce à son aptitude à capturer

---

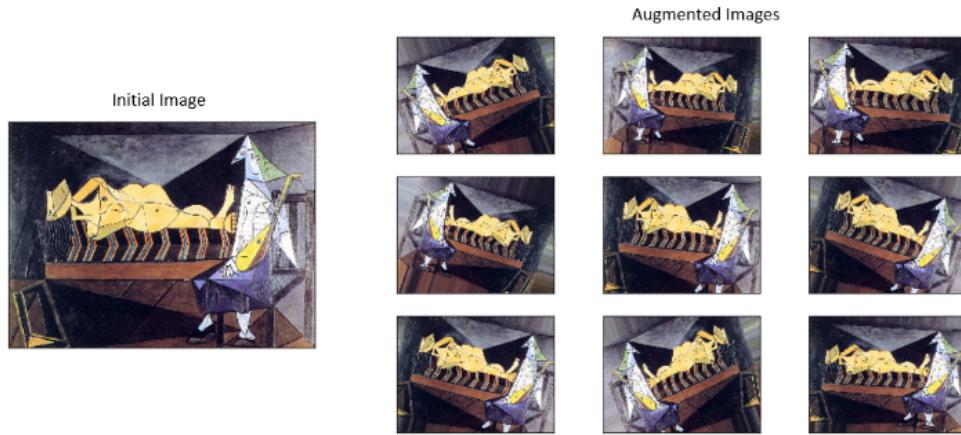
des représentations hiérarchiques et abstraites des données. Parmi les architectures neuronales les plus populaires pour le traitement d'images, on trouve les réseaux de convolutions (Convolutional Neural Networks - CNN). Un réseau CNN typique comprend successivement des couches de convolution, activation (ReLU), pooling (downsampling), dropout, batch normalization et fully connected layers. Ces blocs permettent d'extraire progressivement des features de plus en plus complexes et discriminantes.

## **2. Preprocessing des Images**

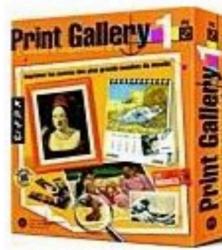
En vue d'améliorer, normaliser et adapter les images aux besoins spécifiques des algorithmes d'apprentissage automatique, il est nécessaire d'appliquer certaines transformations aux images afin de rendre notre modèle plus robuste et capable de généraliser à des images qui ne sont pas exactement identiques à celles utilisées pendant l'entraînement.

Utilisant la classe « `ImageData Generator` » de la bibliothèque Keras de Tensorflow, nous y passons les options suivantes :

- ❖ **`rotation_range=10`** : Cette option fait pivoter chaque image autour de son centre de gravité, avec un angle compris entre -10 et +10 degrés.
- ❖ **`width_shift_range=0.1 & height_shift_range=0.1`** : Ces deux options font respectivement glisser l'image horizontalement et verticalement, avec un facteur multiplicatif choisi uniformément au hasard entre -0.1 et +0.1 fois la largeur et hauteur originale de l'image.
- ❖ **`zoom_range=0.1`** : Cette option zoome de façon aléatoire dans l'image, avec un coefficient de zoom choisi uniformément entre 1-0.1 et 1+0.1
- ❖ **`horizontal_flip=True`** : Cette dernière option inverse horizontalement (de gauche à droite) chaque image avec une probabilité de 50%.



- ❖ **normalisation des Couleurs :** Nous avons ramené les valeurs RGB de chaque pixel dans une plage standardisée (entre 0 et 1) pour homogénéiser les contrastes et intensifier les gradients de couleurs.
- ❖ **normalisation des tailles:** Pour harmoniser la dimensionnalité des images, nous avons renormalisé celles-ci vers une taille fixe en conservant leurs proportions. Cela évite de déformer les objets représentés et garantit une granularité uniforme dans les features extraites. Notons que cette renormalisation varie en fonction des modèles utilisés.
- ❖ **masquage des bords:** Nous observons également que de nombreuses images ont des contours blancs, plus ou moins importants. Afin de « contraindre » le modèle à déterminer les bonnes zones de prédictions, nous décidons de supprimer ces contours.



*Avant prétraitement*

*Après prétraitement*

### **3. Approche frontale: leNet**

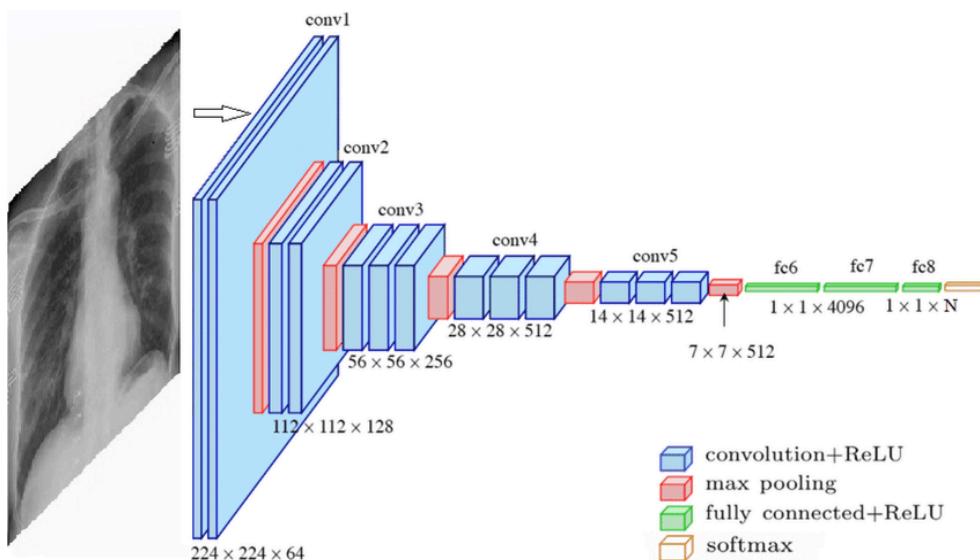
LeNet-5 est un des premiers réseaux de neurones convolutifs. Il s'est illustré lors de la reconnaissance de chiffres manuscrits. Aujourd'hui, LeNet-5 est devenu un CNN simple avec ces sept couches. Nous souhaitions partir de cette architecture pour entraîner un premier modèle. Le Score F1 décevant de 26% avec surapprentissage sur l'ensemble de nos données avec 5 epochs nous a permis cependant de prendre conscience de la richesse des nos images en entrée. Notre quête de modèle et d'architecture plus adaptés ne faisait que commencer !

### **4. Approche Transfer learning**

La modélisation d'images consiste à créer des modèles mathématiques et algorithmiques capables d'analyser et de comprendre les propriétés visuelles des images. Elle vise à extraire des informations pertinentes à partir des pixels constituant une image numérique afin de faciliter diverses applications telles que la reconnaissance d'objets, la classification d'images, la segmentation d'image, la restauration d'image, etc..

Dans le cadre du projet, nous construirons un réseau de neurones convolutif (Convolutional Neural Network - CNN) à l'aide du package Keras, réseau construit pour traiter de la donnée de type image. Les couches de convolution sont aujourd'hui vues comme les meilleurs extracteurs de features pour des problèmes de classification liés à l'image.

Nous utiliserons plus particulièrement l'architecture du modèle VGG16 incluant 13 couches convolutionnelles, cinq couches maximales de pooling et trois couches complètement connectées (FC).



---

Nous comparerons les résultats obtenus avec ceux d'un modèle de réseau neuronal profond qui a été proposé en 2019 par Mingxing Tan et Quoc V. à savoir le modèle EfficientNetB1. L'objectif principal derrière ce modèle était d'améliorer l'efficience globale des réseaux de neurones profonds en augmentant leur profondeur, leur largeur et leur résolution simultanément, sans ajouter trop de paramètres supplémentaires. Il dispose d'une architecture modulaire divisée en six groupes de blocs de couches convolutionnelles, chaque groupe comportant une série de blocs identiques avec un facteur d'expansion progressif. Chaque bloc de couches contient une combinaison de couches de convolution, de normalisation batch et d'activation ReLU. Entre les groupes de blocs, une couche de mise à l'échelle est placée pour adapter la résolution de l'image à chaque niveau de profondeur accru.

L'apprentissage d'un modèle à l'aide d'un réseau de neurones implique plusieurs étapes, dont voici une description détaillée :

- ❖ **Acquisition des données** : La première étape du processus d'apprentissage consiste à collecter et organiser les données d'entraînement. Ici, nous récupérons les chemins d'accès de nos images que nous insérons dans un dataframe contenant également les labels liés.
- ❖ **Prétraitement des données** : À l'aide de la classe ImageDataGenerator, nous effectuons le pré-processing sur les images chargées. Nous les redimensionnons à une taille prescrite pour notre modèle. Nous en profitons également pour créer un dataset d'entraînement utilisé pour actualiser les poids du modèle, un dataset de validation pour affiner les hyperparamètres et éviter le surapprentissage, et un dernier dataset de test pour évaluer les performances du modèle.
- ❖ **Division en lots** : Pour faciliter l'apprentissage, les données sont divisées en mini-lots, qui sont traités par le modèle en une seule itération. Les mini-lots permettent d'utiliser les algorithmes d'optimisation stochastique et d'accélérer le processus d'actualisation des poids.
- ❖ **Choix de l'optimiseur** : L'optimiseur joue un rôle central dans le processus d'apprentissage. Il agit sur les gradients pour minimiser la fonction de perte. Nous utilisons l'optimiseur **ADAM** (Adaptive Moment Estimation), algorithme d'optimisation qui prend en compte l'histoire des gradients passés pour adapter le taux d'apprentissage aux différents paramètres du modèle.

- 
- ❖ **Choix de la fonction de perte** : fonction mathématique qui mesure l'écart entre la prédiction du modèle et la vérité. Elle guide l'apprentissage en indiquant au modèle dans quelle direction il doit ajuster ses paramètres pour minimiser cette distance. La fonction de perte **categorical crossentropy**, que nous avons utilisé, est une fonction de perte couramment utilisée pour la classification multi-classes, où les prédictions sont représentées sous forme de probabilités pour chaque classe. Elle calcule la divergence de Kullback-Leibler entre les probabilités réelles et les probabilités prédites, ce qui revient à comparer les distributions de probabilités.
  - ❖ **Choix de la métrique de performance** : L'accuracy est une métrique de performance utilisée pour évaluer la qualité d'un modèle de classification. Elle représente la fraction de prédictions correctes parmi l'ensemble des prédictions effectuées. Autrement dit, l'accuracy mesure le pourcentage de prédictions pour lesquelles le modèle a effectivement prédit la bonne classe. Ceci paraît tout à fait opportun.
  - ❖ **Évaluation** : Enfin, le modèle entraîné est évalué sur les données de test pour déterminer ses performances réelles. Les métriques d'évaluation que nous mettons en avant sont l'**accuracy** du modèle et, ayant constaté un déséquilibre important de classes, le **f1\_score ‘weighted’**. Il s'agit d'une métrique qui calcule le F1-score séparément pour chaque classe, puis effectue une moyenne pondérée de ces scores en tenant compte du support de chaque classe (c'est-à-dire le nombre d'échantillons appartenant à chaque classe). Cette approche permet de prendre en compte le déséquilibre de classes potentiel dans le jeu de données, et d'obtenir une estimation plus robuste de la performance globale du modèle.

## A. Modèle VGG16

Voici le détail du modèle que nous avons utilisé :

Layer (type)	Output Shape	Param #
input_layer_1 (InputLayer)	(None, 224, 224, 3)	0
vgg16 (Functional)	(None, 7, 7, 512)	14,714,688
global_average_pooling2d (GlobalAveragePooling2D)	(None, 512)	0
dense (Dense)	(None, 1024)	525,312
batch_normalization (BatchNormalization)	(None, 1024)	4,096
dropout (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524,800
batch_normalization_1 (BatchNormalization)	(None, 512)	2,048
dense_2 (Dense)	(None, 27)	13,851

Nous avons ajouté les layers suivants au modèle VGG16 à savoir :

- ❖ **Global Average Pooling 2d** est un type d'opération de regroupement. Les dimensions spatiales du tenseur d'entrée sont réduites à 1x1 en prenant la valeur moyenne de tous les éléments le long de ces dimensions pour chaque carte des caractéristiques. Cela donne lieu à un tenseur de sortie avec une forme (*batch\_size, num\_filters, 1, 1*). Le tenseur résultant peut ensuite être aplati et passé à travers une ou plusieurs couches entièrement connectées avant d'être introduit dans la fonction d'activation softmax pour les tâches de classement. Elle contribue à prévenir le surapprentissage en réduisant le nombre de paramètres du modèle.
- ❖ **Dense:** couche de réseau neuronal artificiel où chaque neurone reçoit des connexions depuis tous les neurones présents dans la couche précédente. Ces connexions ont des poids et des biais appris pendant l'entraînement, ce qui signifie qu'ils ont des poids et des biais modifiables lors de l'apprentissage. Elle est utilisée pour classifier des vecteurs de données à haute dimensionnalité.
- ❖ **Batch Normalization:** la normalisation par lot est une technique de régularisation qui vise à accélérer l'apprentissage et améliorer la performance des réseaux de neurones profonds. Pour chaque mini-lot de données, elle calcule la moyenne et la variance empiriques de l'activation de chaque neurone, puis effectue une normalisation affine afin de conserver la capacité expressive du réseau. Elle favorise aussi la convergence rapide des gradients pendant l'entraînement, diminuant ainsi le temps requis pour atteindre une bonne solution.

- 
- ❖ **Dropout:** technique de régularisation destinée à minimiser le risque de surapprentissage et augmenter la robustesse des réseaux de neurones artificiels. Il agit en désactivant aléatoirement un sous-ensemble de neurones durant l'entraînement, forçant ainsi le reste du réseau à compenser cette perte d'information. Ainsi, les poids de chaque neurone ne deviennent pas trop spécialisés ni excessivement adaptés aux particularités du jeu de données d'entraînement, ce qui conduit à une meilleure généralisation lorsqu'il s'agira de réaliser des prédictions sur des exemples inconnus.

## B. Modèle EfficientNetB1

Voici le détail du modèle que nous avons utilisé :

Layer (type)	Output Shape	Param #
input_layer_1 (InputLayer)	(None, 224, 224, 3)	0
efficientnetb1 (Functional)	(None, 7, 7, 1280)	6,575,239
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dense (Dense)	(None, 1024)	1,311,744
batch_normalization (BatchNormalization)	(None, 1024)	4,096
dropout (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524,800
batch_normalization_1 (BatchNormalization)	(None, 512)	2,048
dense_2 (Dense)	(None, 27)	13,851

## C. Modèle EfficientNetV2L

L'architecture EfficientNetV2L proposée par Google Research fait partie des derniers développements en termes de modèles de vision par ordinateur. Elle améliore encore davantage les principes fondamentaux mis en place dans les précédentes versions telles que EfficientNet, EfficientNet-EdgeTPU et EfficientNetV2, en combinant diverses avancées réalisées dans des domaines tels que la recherche d'architecture neuronale, les stratégies de formation et les méthodes d'évolutivité. Grâce à ses performances exceptionnelles associées à une efficacité computationnelle toujours optimale, EfficientNetV2L rivalise désormais avec d'autres grandes architectures connues.

Voici le détail du modèle que nous avons utilisé :

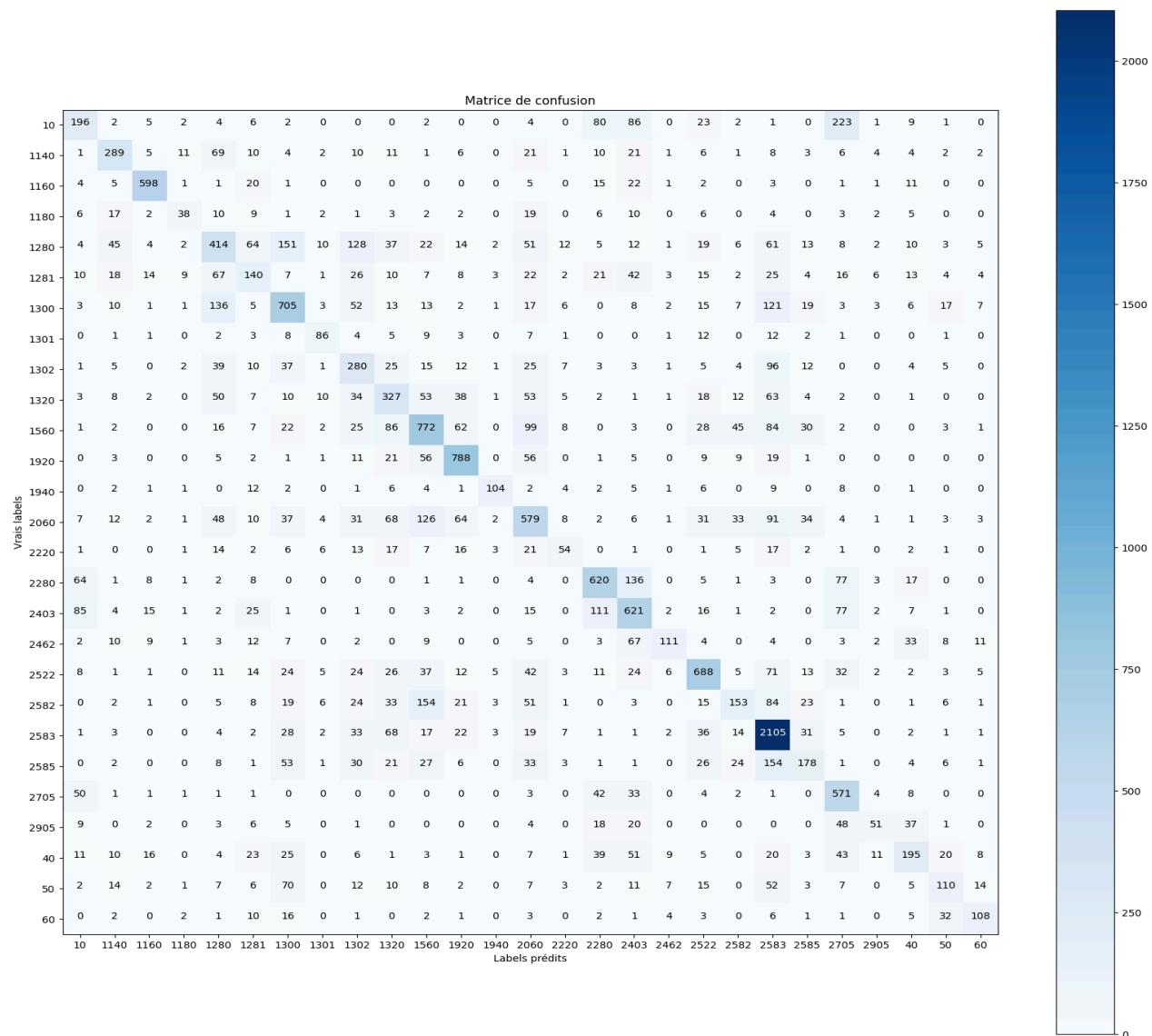
Layer (type)	Output Shape	Param #
input_layer_1 (InputLayer)	(None, 380, 380, 3)	0
efficientnetv2-l (Functional)	(None, 12, 12, 1280)	117,746,848
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
dense (Dense)	(None, 1024)	1,311,744
batch_normalization (BatchNormalization)	(None, 1024)	4,096
dropout (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 512)	524,800
batch_normalization_1 (BatchNormalization)	(None, 512)	2,048
dense_2 (Dense)	(None, 27)	13,851

## D. Scores obtenus

<u>MODÈLE</u>	<u>ACCURACY</u>	<u>F1SCORE WEIGHTED</u>	<u>DÉLAI</u>
VGG16	63.8 %	63.2 %	10 ms/step
EFFICIENTNET_B1	53.1 %	53.7 %	9 ms/step
EFFICIENTNET_V2	62 %	61 %	46 ms/s

## Heatmap des matrices de confusion:

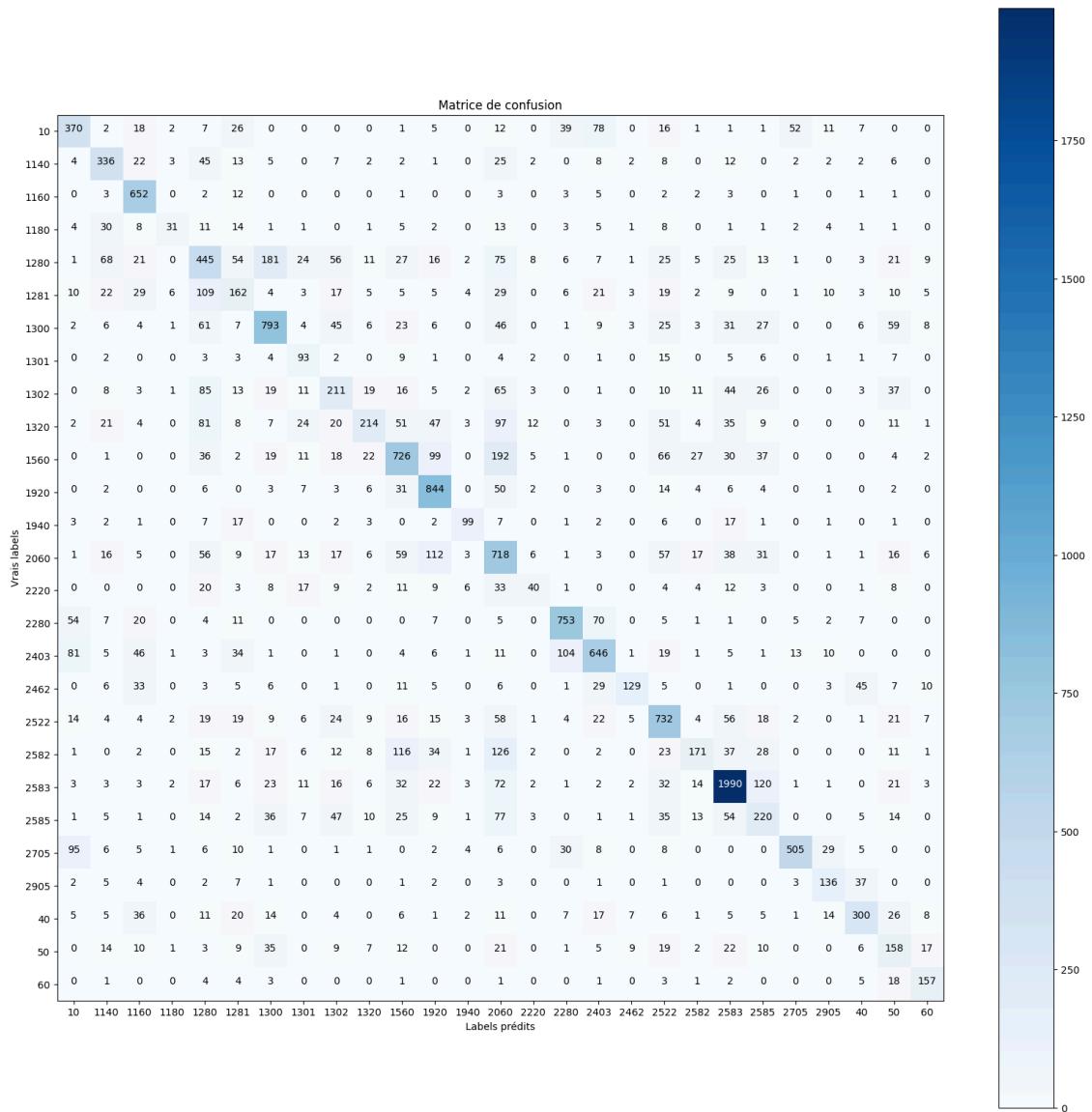
**Modèle VGG16**



Voici les classes ayant une accuracy inférieure à 50%:

- 1280: Jouet, Enfant, Puériculture > OCCASIONS
- 1281: Jouet, Enfant, Puériculture > JEU ÉDUCATIF ET INTERACTIF
- 1301: Jouet, Enfant, Puériculture > JEUX DE CAFE & TEXTILE PUERICULTURE
- 1302: Jouet, Enfant, Puériculture > JEUX DE PLEIN AIR
- 2060: Maison > DECORATION
- 2220: Brico, Jardin, Animalerie > ANIMALERIE
- 50: Jeux vidéo, Console > ACCESSOIRES

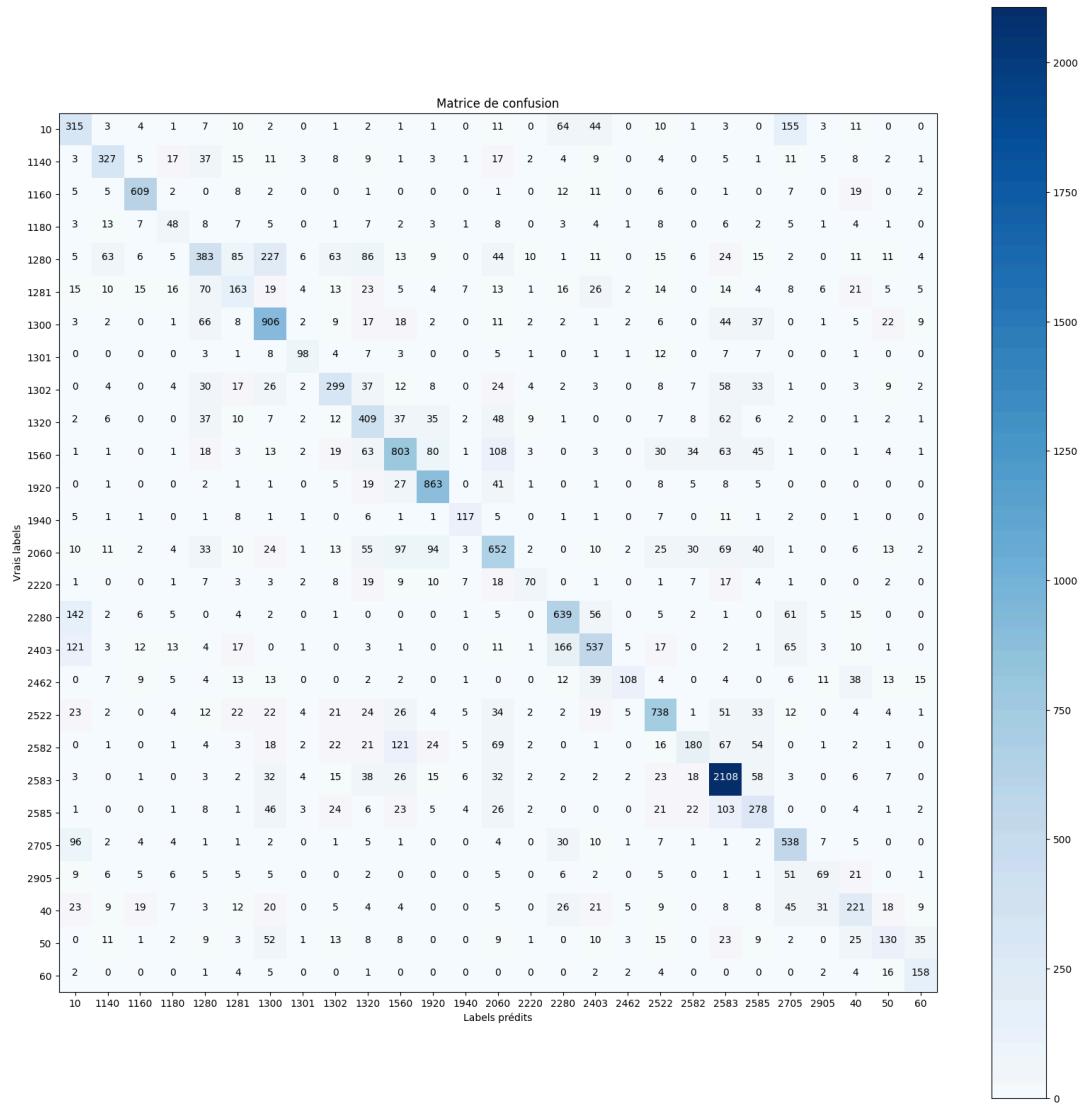
## Modèle EFFICIENTNET\_B1



Voici les classes ayant une accuracy inférieure à 50%:

- 1280: Jouet, Enfant, Puériculture > OCCASIONS
- 1281: Jouet, Enfant, Puériculture > JEU ÉDUCATIF ET INTERACTIF
- 1301: Jouet, Enfant, Puériculture > JEUX DE CAFÉ & TEXTILE PUÉRICULTURE
- 1302: Jouet, Enfant, Puériculture > JEUX DE PLEIN AIR
- 2060: Maison > DÉCORATION
- 2220: Brico, Jardin, Animalerie > ANIMALERIE
- 2585: Brico, Jardin, Animalerie > BRICOLAGE
- 50: Jeux vidéo, Console > ACCESSOIRES

## Modèle EFFICIENTNET\_V2



Voici les classes ayant une accuracy inférieure à 50%:

- 10: Livre > FOURNITURE ET MANUEL SCOLAIRE
- 1180: Jouet, Enfant, Puériculture > DÉGUISEMENT & MAGIE
- 1281: Jouet, Enfant, Puériculture > JEU ÉDUCATIF ET INTERACTIF
- 1320: Jouet, Enfant, Puériculture > PUERICULTURE
- 2585: Brico, Jardin, Animalerie > BRICOLAGE
- 2905: Jeux vidéo, Console > SECONDE MAIN
- 40: Jeux vidéo, Console > ACCESSOIRES

---

Nous concluons des heatmaps et des rapports de classification de nos différents modèles que certaines classes sont très difficiles à identifier pour nos algorithmes, tout aussi puissants et robustes qu'ils soient. Il est cependant compliqué de leur en vouloir car distinguer par exemple un accessoire de gaming de seconde main d'un autre neuf peut s'avérer un exercice parfois hasardeux, même pour un humain. La multiplicité des classes et les frontières parfois extrêmement minces entre elles nous empêchent d'obtenir un taux de bonnes prédictions dépassant les 65%, contrairement aux modèles sur le texte qui avoisinent assez facilement les 80%.

Fort de ce constat, il semble intéressant dans notre situation de visualiser la prédiction sur nos images non pas comme un but en soi, mais comme un atout qui va nous aider à potentiellement améliorer les prédictions effectuées sur le texte, nous amenant à nous intéresser aux modèles multimodaux.

## **5. Interprétabilité via LIME**

### ❖ **Méthodologie**

Pour l'interprétabilité des images, nous avons utilisé la librairie lime (Local Interpretable Model-agnostic Explanations). L'interprétation se base sur une fonction de segmentation de l'image, les segments définis sont appelés superpixels. Cette segmentation en superpixels a été réalisée avec la méthodologie quickshift du package skimage.segmentation. Chaque superpixel est défini comme une région prenant en compte la luminosité ainsi que les couleurs des pixels de l'image. Cette segmentation est représentée par l'image "superpixels" dans les interprétations suivantes. La classification est réalisée sur chacun des superpixels pour définir une prédiction d'ensemble en sommant les différentes prédictions.

### ❖ **Limitations**

Nous identifions principalement deux limites par rapport aux interprétations réalisées avec cette librairie. La première réside dans la nature de la librairie qui se concentre sur une interprétation locale du modèle (les superpixels). Cette librairie n'interprète donc

---

pas l'image dans son ensemble et ne prend pas en compte le contexte de l'image interprétée. Ce biais peut être plus ou moins fort en fonction du découpage de l'image en superpixel et du preprocessing de l'image. Dans notre cas, il est possible que les bords blancs identifiés sur certaines images aient un impact sur l'interprétation réalisée par cette librairie.

La seconde tient dans les ressources utilisées pour réaliser cette interprétation. Effectuer l'interprétation sur chaque super pixel demandera l'utilisation de plus de ressources et limitera l'utilisation de la librairie sur une grande quantité d'images.

## A. Interprétation 1: interprétation multiple

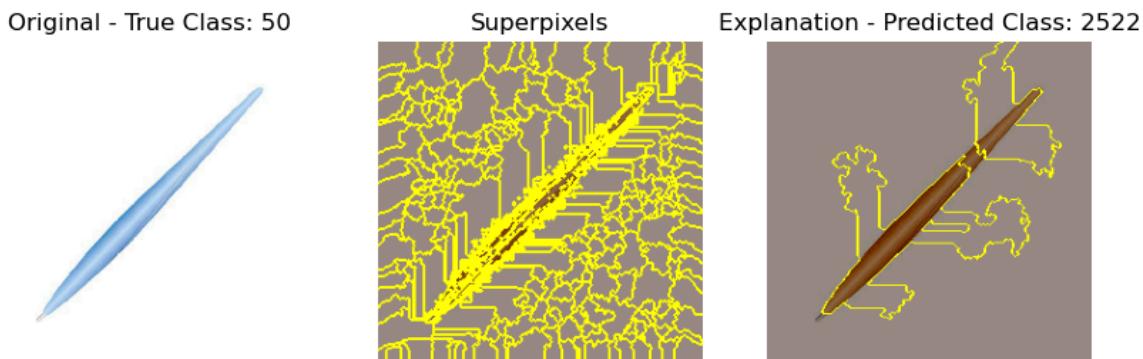
La prédiction de la classe de l'image ci-dessous est erronée.

Le modèle prédit une classe 2522 (que nous avons définie comme "Livre > FOURNITURES PAPETERIE"), tout comme le modèle de régression logistique pour le texte au lieu de la classe 50 (que nous avons défini comme "Jeux vidéo, Console > ACCESSOIRES"). Cela est lié au fait que l'image comme le texte pour cet objet, se rapproche très fortement de la classe 2522.

L'image "Explanation - predicted class" représente les features (superpixels) les plus déterminantes pour la classification et n'affiche que celles associées à la classe prédite.

```
top class: 2522 - prediction: 0.5153385
top class 2: 2060 - prediction 2: 0.09916976
top class 3: 1560 - prediction 3: 0.074299864
top class 4: 2585 - prediction 4: 0.058678128
top class 5: 1300 - prediction 5: 0.047194622
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



## **B. Interprétation 2: interprétation correcte**

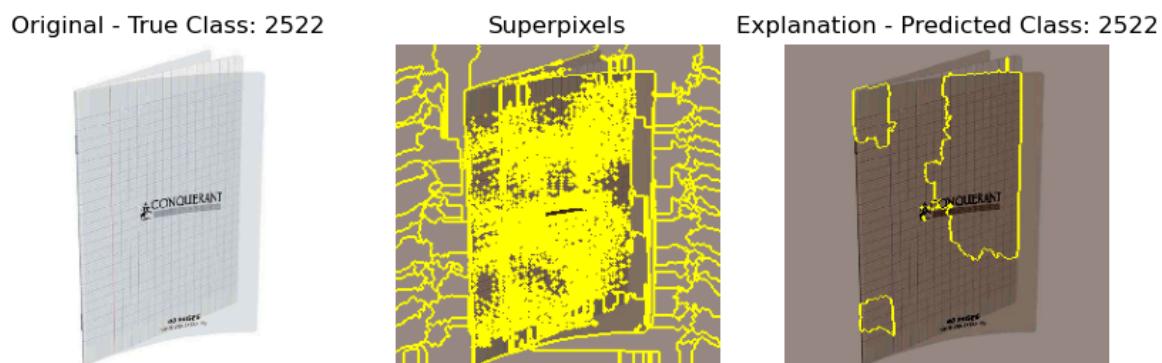
La prédiction de la classe de l'image ci-dessous est correcte.

Le modèle prédit une classe 2522 (que nous avons définie comme “Livre > FOURNITURES PAPETERIE”), tout comme le modèle de régression logistique avec une précision de 99,7%.

L'image “Explanation - predicted class” représente les features (superpixels) les plus déterminantes pour la classification et n'affiche que celles associées à la classe prédite.

```
top class: 2522 - prediction: 0.9976568
top class 2: 2585 - prediction 2: 0.0012302345
top class 3: 2583 - prediction 3: 0.0006622806
top class 4: 1560 - prediction 4: 0.00014531083
top class 5: 2582 - prediction 5: 8.4919746e-05
```

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).



## **6. Interprétabilité via Grad-CAM**

### ❖ Méthodologie

Pour enrichir nos approches sur l'interprétabilité de modèles sur les images, nous avons instrumentalisé nos modèles afin d'obtenir des cartes de gradient des images. Contrairement à Lime qui interprète les prédictions en créant des versions modifiées de l'image d'origine, en faisant des prédictions sur ces images modifiées, en apprenant un modèle linéaire simple pour approximer le comportement du modèle autour de l'image d'origine, Grad-CAM, en revanche, utilise les gradients de la sortie de la classe cible par rapport aux caractéristiques d'une couche convulsive pour produire une carte de chaleur indiquant les régions importantes de l'image. Cette technique a été expérimentée sur deux modèles. La proximité de score F1 du

modèle VGG16 et EfficientNet nous ont en effet surpris. VGG16 est une architecture plus ancienne et plus simple, avec 16 couches de profondeur. Nous avons entraîné un modèle basé sur EfficientNetB4, plus profond et plus large que VGG16. En général, EfficientNetB4 devrait surpasser VGG16 en termes de précision sur la plupart des tâches de classification d'images.

## ❖ Résultats

Les résultats ont confirmé des sensibilités différentes entre les deux modèles ainsi qu'une capacité à l'avantage de EfficientNet de se focaliser sur des éléments plus pertinents que VGG16. Par exemple, l'application de Gram-CAM sur une couverture de livre extrait de l'étiquette de code-produit 10 révèle en effet que EfficientNet se concentre sur la capture des zones de texte alors que les zones d'intérêt de VGG16 sont plus difficilement identifiables.



## 7. Conclusion

Nous avons abordé la classification unimodale des images de manière progressive. Suite à un prétraitement des images pour extraire les informations pertinentes, nous avons constaté que l'architecture LeNet ne fournissait pas de résultats satisfaisants. Cela nous a naturellement orientés vers l'apprentissage par transfert d'apprentissage, en utilisant les modèles VGG16, EfficientNetB1 et EfficientNetV2L. Bien que les scores F1 obtenus avec chaque modèle soient approximativement de 60%, l'interprétabilité des modèles a révélé des différences significatives dans leurs zones d'intérêt différencierées par étiquette. Pour une meilleure compréhension des prédictions, nous avons employé LIME, permettant d'obtenir des interprétations multiples et précises. Enfin, l'utilisation de Grad-CAM nous a permis de visualiser les régions de l'image ayant le plus influencé les prédictions du modèle.

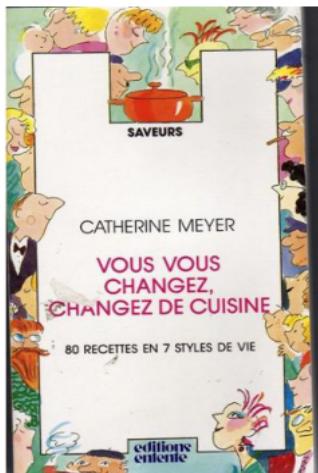
## V. Classification multimodale des images

### 1. Généralités

La classification multimodale intègre simultanément plusieurs types de données comme des images et des textes afin d'améliorer les prédictions sur les étiquettes communes en apportant au machine Learning une compréhension relationnelle plus riche et plus complète. Les enjeux sont variés, allant de la simple correspondance multilatérale des données basée sur l'étiquette, à la synchronisation temporelle des données, jusqu'à la création d'un espace de représentation commun pour différentes modalités afin de saisir les similarités sémantiques.

En guise d'introduction, nous allons considérer un modèle multimodal naïf qui juge opportun de convertir la modalité image en modalité texte, puisque cette dernière affiche de meilleurs résultats. Cet exercice est rendu possible au moyen de la librairie en accès libre Pytesseract qui précisément permet d'extraire et retranscrire le texte contenu dans des images. Voici la transformation subie par deux images provenant d'annonces :

Texte extrait :CATHERINE MEYER vous VOUS CHANGEZ, 4HANGEZ DE CUISIN!

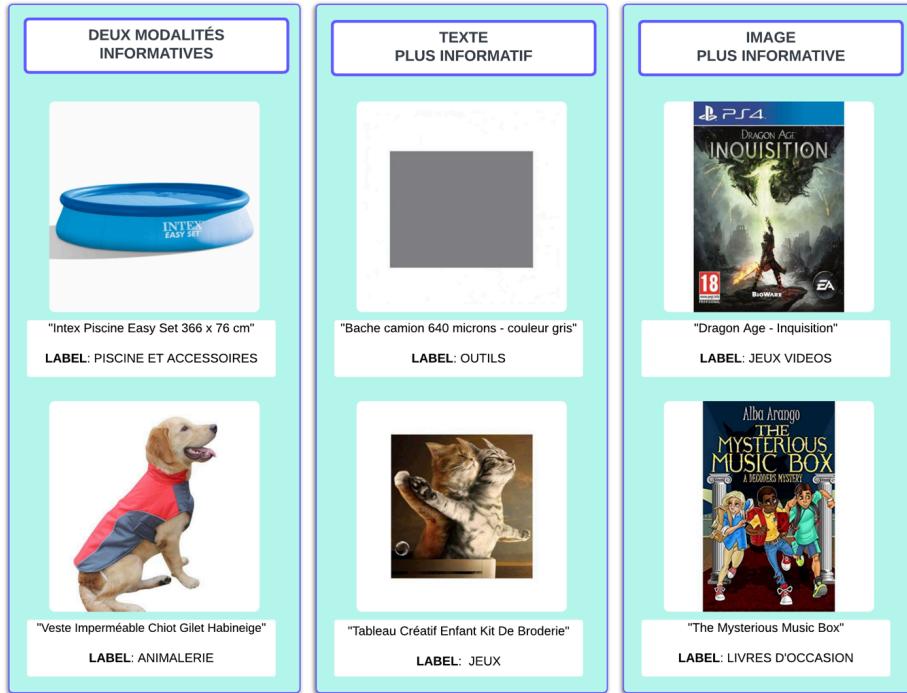


Texte extrait :1/2" x19mm 3/8" x14 1/4" x1 4m



Cet exemple met en lumière le défi majeur que constitue l'harmonisation des modalités. En effet, loin d'améliorer les performances, cette transposition simple du multimodal en unimodal ne fait globalement qu'ajouter du bruit et dégrade nettement les scores obtenus avec le texte seul. Il faut d'abord éviter les interférences négatives entre modalités, tout en prenant garde à ne pas négliger complètement un mode pour l'autre (phénomène de modality collapse)

et réussir enfin à révéler les relations productives entre les différentes natures de données. Nous illustrons ce principe avec quelques cas de figure présents dans le dataset :



Il existe principalement deux façons de combiner les différentes modalités dans une tâche de classification multimodale : la fusion précoce et la fusion tardive.

La fusion précoce consiste à combiner les différentes modalités dès le début du processus d'apprentissage automatique. L'avantage de cette technique, qui fusionne directement les caractéristiques des différentes modalités, est de permettre au modèle d'inférer des relations de généralisation communes grâce aux interactions induites entre les modalités lors de l'apprentissage. Cette stratégie de fusion offre une efficacité computationnelle car elle ne nécessite qu'un seul apprentissage pour toutes les modalités. Cependant, l'exploitation précoce des interactions intermodales peut rendre l'apprentissage très sensible à la propagation d'erreurs et aux modalités bruitées. Un autre inconvénient est le manque de flexibilité pour ajuster la fusion à chaque modalité individuellement. Ce manque de flexibilité peut même rendre la fusion inefficace si une modalité est beaucoup plus informative qu'une autre, entraînant un effondrement des modalités et concentrant l'apprentissage uniquement sur l'une d'elles. Enfin, cette technique peut être difficile à mettre en œuvre si les modalités présentent des données manquantes croisées.

---

En revanche, la fusion tardive implique l'entraînement de modèles d'apprentissage automatique distincts pour chaque modalité, suivis de la combinaison de leurs prédictions pour produire une prédition finale. Cette prédition finale est donc réalisée au niveau des décisions en sortie de chaque modèle. L'avantage de cette technique est sa robustesse face aux erreurs et aux modalités bruitées, ainsi que sa flexibilité de combinaison entre les modalités. L'inconvénient de cette technique est son coût computationnel, car elle nécessite autant de modèles à entraîner qu'il y a de modalités à combiner, et sa capacité limitée à capturer les interactions riches entre les modalités traitées séparément.

Enfin, ces techniques se combinent avec les autres techniques vu précédemment tels que le transfer Learning ou le preprocessing.

## **2. Nos différentes approches**

Dans notre projet, nous avons mis en place ces deux techniques :

### **A. Fusion précoce**

La fusion précoce est une technique utilisée en apprentissage profond multimodal où les données provenant de plusieurs modalités sont combinées à un stade précoce, généralement avant que les données ne soient introduites dans un modèle pour l'entraînement. Voici une explication détaillée de la fusion précoce :

Dans la fusion précoce, les données brutes ou les représentations des caractéristiques de différentes modalités sont combinées pour former un seul vecteur d'entrée. Cette combinaison peut se produire au niveau des données brutes ou après une étape initiale d'extraction des caractéristiques. Dans notre scénario, les caractéristiques textuelles et les caractéristiques visuelles sont concaténées en un seul vecteur de caractéristiques.

Nous énumérons ci-dessous cette quelques avantages que présente cette technique :

- ❖ **Interactions riches entre les caractéristiques** : En combinant les caractéristiques tôt, le modèle peut apprendre des interactions et des corrélations complexes entre différentes modalités, ce qui peut conduire à de meilleures performances sur des tâches qui bénéficient d'informations multimodales intégrées.
- ❖ **Pipeline de traitement unifié** : Le pipeline de traitement entier est unifié, simplifiant l'architecture puisqu'un seul modèle doit être entraîné et maintenu.

---

### **a) Bert +LSTM (texte) + InceptionV3 (image)**

BERT (Bidirectional Encoder Representations from Transformers) est un modèle pré-entraîné sur un vaste corpus de texte qui prend en compte le contexte des mots dans une phrase. Cela lui permet de capturer des nuances subtiles de signification dans le texte, ce qui est crucial pour des tâches de classification où la sémantique et le contexte sont importants. BERT a établi de nouveaux records de performance dans de nombreuses tâches de traitement du langage naturel (NLP), y compris la classification de texte, sur une variété de jeux de données. Son architecture transformative bidirectionnelle lui permet de capturer des informations contextuelles sur les mots, ce qui se traduit par des performances exceptionnelles dans la plupart des cas.

LSTM, qui signifie Long Short-Term Memory (Mémoire à Long et Court Terme), est un type d'architecture de réseau de neurones récurrents (RNN) conçu pour capturer et modéliser efficacement les dépendances temporelles et les corrélations à long terme dans les données séquentielles. Les LSTM sont particulièrement utiles pour les tâches où le contexte et l'ordre des points de données sont cruciaux, comme la prévision de séries temporelles, le traitement du langage naturel et la reconnaissance vocale.

Les LSTM offrent de nombreux avantages pour la classification des textes, grâce à leur capacité à traiter et à comprendre les séquences de données. Voici les principaux avantages des LSTM dans ce domaine :

- ❖ **Capture des dépendances à long terme:** Les LSTM sont particulièrement efficaces pour capturer les dépendances à long terme dans les séquences de texte. Cela signifie qu'ils peuvent comprendre le contexte global d'un document, même si des informations cruciales apparaissent loin les unes des autres dans le texte. Par exemple, ils peuvent relier une référence faite au début d'un texte à une information donnée beaucoup plus tard, ce qui est essentiel pour une compréhension précise.
- ❖ **Gestion du problème de gradient:** Les LSTM atténuent les problèmes de gradient qui disparaît et de gradient explosif, communs dans les réseaux de neurones récurrents traditionnels. Cela permet aux LSTM de conserver et d'utiliser efficacement les informations sur des séquences textuelles longues, améliorant ainsi les performances de classification.

- 
- ❖ **Conservation des informations pertinentes:** Grâce à leurs portes d'entrée, de sortie et d'oubli, les LSTM peuvent apprendre à filtrer les informations pertinentes et à ignorer les informations non pertinentes dans un texte. Cela améliore la capacité du modèle à se concentrer sur les aspects importants du texte pour la classification.
  - ❖ **Modélisation contextuelle:** Les LSTM sont capables de capturer le contexte d'un mot ou d'une phrase au sein d'un texte, ce qui est crucial pour la classification des textes. Par exemple, le mot "banc" peut avoir des significations différentes selon le contexte ("banc de poissons" vs "s'asseoir sur un banc"). Les LSTM peuvent comprendre et différencier ces contextes grâce à leur architecture.

InceptionV3 est quant à lui un célèbre modèle de Deep Learning destiné à la reconnaissance visuelle, développé et rendu public par Google Research. Cette version marquante de la famille Inception tire profit de nombreuses idées innovantes présentées dans des articles universitaires, notamment GoogLeNet (InceptionV1), et combine intelligemment celles-ci dans un unique cadre. Depuis sa publication en 2015, InceptionV3 continue de servir de point de départ à des milliers de projets axés sur la vision artificielle, offrant un bon équilibre entre complexité et performance.

#### **Architecture du modèle utilisé:**

Le modèle de classification de textes utilise le modèle BERT Base, l'intégrant avec LSTM pour exploiter efficacement le contexte textuel à court et long terme, une approche hybride pour capturer la sémantique nuancée des textes. L'architecture Inception V3 a été adaptée à notre tâche spécifique, en exploitant le concept de Transfer Learning. En utilisant des poids pré-entraînés du jeu de données ImageNet, nous visons à améliorer l'efficacité de l'apprentissage et la précision initiale.

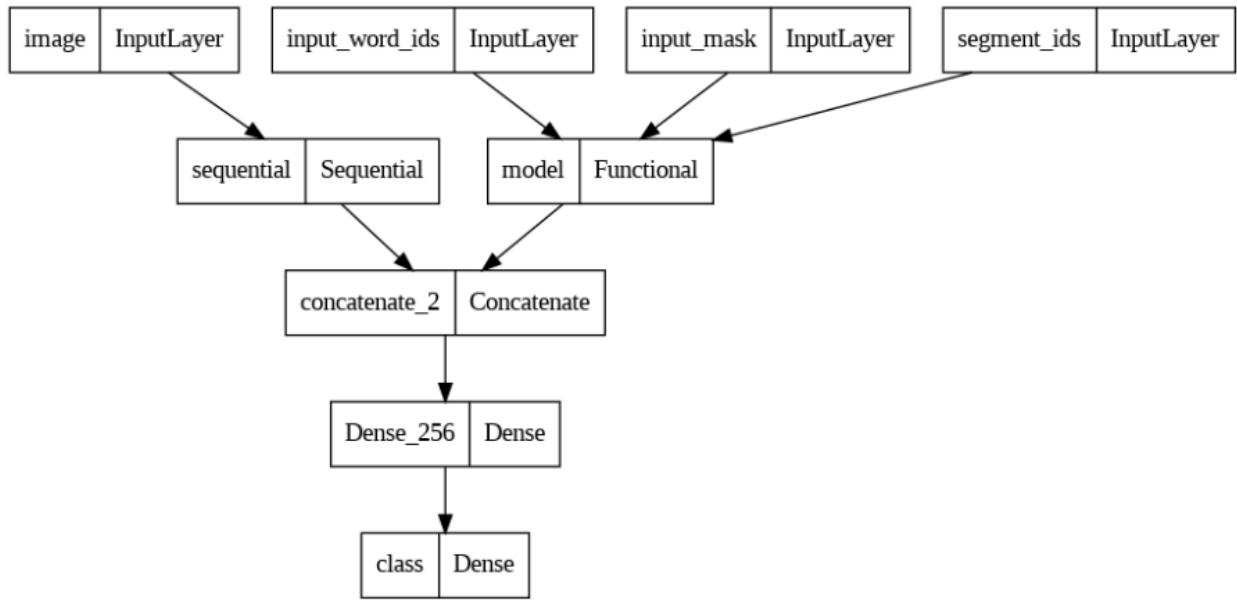
- **Modèle BERT Base :** Optimisé pour la performance avec 12 couches et une couche cachée de 768 dimensions, le modèle BERT prétraite les textes d'entrée pour les transformer en un format adapté à l'apprentissage profond.
- **Intégration LSTM :** Une couche LSTM de 128 unités suit la sortie de BERT, choisie pour son équilibre entre efficacité computationnelle et performance. Cela est encore amélioré avec une couche dense de 256 unités et plusieurs couches de dropout pour gérer de manière robuste les données textuelles diverses et

---

minimiser le surapprentissage.

- **InceptionV3:** Les deux dernières couches du modèle Inception V3 sont remplacées pour l'adapter au jeu de données Rakuten Challenge. Cela comprend l'ajout d'une couche de pooling moyenne de 8x8, une couche de dropout avec une probabilité de 40% pour prévenir le surapprentissage, et une dernière couche dense softmax avec 27 unités correspondant aux classes du jeu de données.
- **Couche de concaténation :** Les caractéristiques extraites à la fois du modèle Inception (caractéristiques d'image) et du modèle BERT+LSTM (caractéristiques de texte) sont ensuite fusionnées à l'aide d'une couche de concaténation. Cette fusion crée un vecteur unique de caractéristiques combinées, représentant les deux modalités.
- **Traitement post-fusion :** Après la fusion, le vecteur de caractéristiques combiné passe à travers des couches supplémentaires pour préparer la classification
- Une couche dense de 256 unités suivie d'une fonction d'activation ReLU améliore la capacité à capturer des combinaisons non linéaires des caractéristiques fusionnées.
- Une dernière couche dense de 27 unités (correspondant aux 27 classes différentes) utilise une fonction d'activation softmax. Cette couche produit les probabilités à travers les 27 classes, fournissant la décision finale de classification pour l'entrée combinée d'image et de texte.

Voici un résumé graphique du fonctionnement de notre modèle early-fusion:



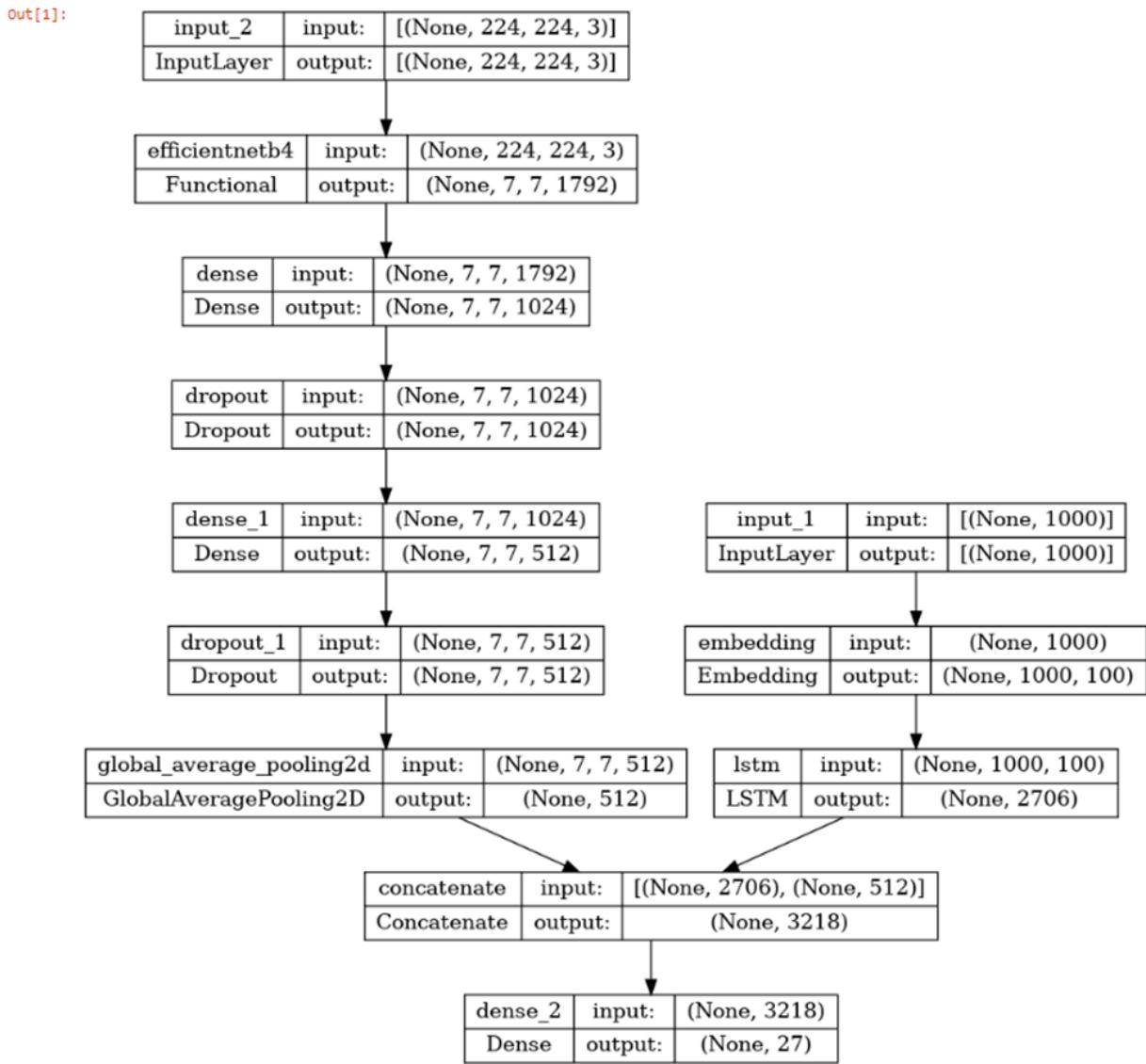
### **b) LSTM (texte) + EfficientNetB4 (image)**

L'objectif de ce modèle simple d'architecture est d'assembler deux modèles image et texte en apprentissage séparé et d'apprécier le gain suite à leur fusion précoce. Nous souhaitons par cette approche extraire des disparités informatives une richesse prédictive et obtenir une base comparative des scores F1 des trois modèles par étiquette.

Pour le traitement des images, nous avons réutilisé le transfert d'apprentissage avec EfficientNetB4 déjà évalué avec le prétraitement des bords blancs. En ce qui concerne le texte, nous avions mis en place un modèle entraîné avec la régression logistique qui ne permettait pas un montage en fusion précoce. Nous avons opté pour un RNN en LSTM en appliquant le même prétraitement. Les Mémoire à Long Terme de Court Terme sont particulièrement efficaces pour le traitement et la génération de texte.

Enfin, nous avons gardé aussi les mêmes outils de traitement comme ImageDataGenerator pour générer et normaliser les lots d'images.

Voici l'architecture obtenue :



## B. Score level fusion

Le score level fusion, aussi appelée decision level fusion, est une méthode de combinaison de prédictions provenant de plusieurs sources (modèles), où les scores de confiance ou probabilités associés à chaque classe candidate sont agrégés afin de produire une décision consolidée. Dans le cadre spécifique de notre projet, il s'agit d'additionner les probabilités par classes obtenues respectivement à partir des modèles BERT (textuel) et EfficientNet V2 (visuel). Les probabilités résultantes de nos modèles ne sont pas mutuellement exclusives, car elles sont générées indépendamment l'une de l'autre, et peuvent donc dépasser ensemble la valeur maximale de 1. Au lieu de les additionner, il existe différentes stratégies ci-dessous détaillées:

- 
- ❖ **Produit naïf:** Multipliez les probabilités A et B, mais cette approche peut rapidement devenir instable avec des probabilités basses.
  - ❖ **Somme pondérée:** Combinez les probabilités A et B en attribuant des coefficients aux probabilités, en tenant compte de leur importance relative. Vous pouvez normaliser le résultat pour qu'il reste compris entre 0 et 1.
  - ❖ **Maximum de vote:** Sélectionnez la probabilité la plus élevée entre A et B. Simple et intuitive, mais elle ne tient pas compte de l'incertitude de chaque probabilité.
  - ❖ **Logarithmes croisés:** Calculez la similarité entre les distributions de probabilités A et B en utilisant les logarithmes croisés. Plusieurs variantes existent, telles que Kullback-Leibler divergence ou Jensen-Shannon distance.
  - ❖ **Mélanges gaussiens:** Approximativez les probabilités A et B avec des mélanges gaussiens, puis combinez-les en fonction de leurs densités de probabilité.

Après plusieurs essais, nous décidons d'utiliser la somme pondérée qui se traduit mathématiquement comme ceci:

Supposons que vous ayez  $P_A$  et  $P_B$ , les probabilités provenant respectivement de l'analyse de l'image et du texte. Nous assignons des poids relatifs  $\alpha$  et  $(1-\alpha)$  aux probabilités, où  $\alpha$  est un coefficient positif tel que  $0 \leq \alpha \leq 1$ . La somme pondérée  $P_{AB}$  sera alors:

$$P_{AB} = \alpha \times P_A + (1-\alpha) \times P_B$$

Ensuite, il nous reste à normaliser  $P_{AB}$  pour qu'il soit compris entre 0 et 1.

Voici quelques-uns des avantages et des inconvénients de cette approche :

#### Avantages :

- ❖ Facile à mettre en œuvre, car elle requiert uniquement l'agrégation des scores de confiance générés par chaque modèle.
- ❖ Pas besoin de re-paramétriser ou d'adapter les architectures originales des modèles source.
- ❖ Utile pour comparer rapidement les performances relatives de différents modèles et observer si leur combinaison apporte des améliorations notables.

---

### **Inconvénients :**

- ❖ Ignore les relations existantes entre les caractéristiques textuelles et visuelles, ce qui peut nuire à la performance globale du système.
- ❖ Ne tire pas profit des synergies potentielles entre les deux domaines, limitant ainsi les gains d'efficacité et d'exactitude.
- ❖ Sensible aux écarts de distribution entre les scores générés par chaque modèle, ce qui peut introduire des biais dans la décision consolidée.

Voici les étapes que nous avons suivies afin d'obtenir nos prédictions:

1. **Formation des Modèles Source** : Nous avons entraîné séparément nos deux modèles source, à savoir BERT pour le traitement du texte et EfficientNet V2 pour la reconnaissance d'images.
2. **Génération des Probabilités par Classes** : Nous avons présenté de nouveaux exemples inconnus aux modèles entraînés et recueilli les probabilités estimées pour chaque classe candidate. Nous nous sommes assurés que les softmax soient correctement appliqués aux sorties logistiques des deux modèles afin d'obtenir des distributions de probabilités cohérentes.
3. **Agrégation des Probabilités** : Nous avons effectué la somme pondérée, élément par élément, les vecteurs de probabilités générés respectivement par BERT et EfficientNet V2. Notons que nous avons pris, après avoir testé plusieurs valeurs, des poids relatifs  $\alpha = 0.2$ . Ce processus a abouti à un unique vecteur de probabilités cumulatives, représentant la contribution relative de chaque classe selon les deux modèles.
4. **Prise de Décision Consolidée** : Nous avons finalement sélectionné la classe ayant la probabilité cumulative la plus élevée comme décision consolidée.

### **C. Résultats obtenus**

Modèle	Type de fusion	Score texte	Score image	Score fusion
Bert + LSTM + InceptionV3	Early	0.628	0.742	0.83
LSTM + EfficientNetB4	Early	0.813	0.606	0.52*
Bert + EfficientNetV2L	Score level	0.848	0.67	0.8704

Etiquette	TEXT	IMAGE	Ecart
	f1-score	f1-score	f1-score
10	0,48	0,61	-0,13
40	0,69	0,65	0,04
50	0,78	0,36	0,42
60	0,9	0,66	0,24
1140	0,77	0,57	0,2
1160	0,91	0,85	0,06
1180	0,63	0,27	0,36
1280	0,7	0,41	0,29
1281	0,57	0,21	0,36
1300	0,95	0,67	0,28
1301	0,94	0,56	0,38
1302	0,83	0,35	0,48
1320	0,8	0,37	0,43
1560	0,83	0,55	0,28
1920	0,92	0,76	0,16
1940	0,85	0,61	0,24
2060	0,8	0,44	0,36
2220	0,86	0,32	0,54
2280	0,72	0,76	-0,04
2403	0,74	0,66	0,08
2462	0,81	0,54	0,27
2522	0,93	0,62	0,31
2582	0,73	0,4	0,33
2583	0,98	0,78	0,2
2585	0,8	0,33	0,47
2705	0,7	0,88	-0,18
2905	0,98	0,86	0,12

\* L'apprentissage en fusion précoce a été limité à 1 epoch sur l'ensemble des 84 800 articles par manque de ressources. La base comparative par étiquette des scores F1 n'a pas pu aboutir afin de mesurer les bénéfices escomptés ou les biais de la fusion précoce. Nous donnons cependant ici les écarts par étiquettes entre les deux modèles séparés texte et images. Nous surlignons en vert le score F1 à l'avantage des images et en bleu un score F1 à l'avantage du texte.

---

### **3. Conclusion**

Les deux méthodes de fusion ont contribué à une amélioration substantielle du score F1 obtenu séparément par la modélisation texte ou image. La fusion tardive, qui sélectionne les probabilités les plus élevées entre les deux modèles image et texte, permet intrinsèquement d'obtenir des prédictions plus robustes. Cette approche semble particulièrement pertinente dans le contexte Rakuten, où selon l'article une modalité peut être nettement plus informative que l'autre.

Par exemple, pour le code-produit 10 (Livre > FOURNITURE ET MANUEL SCOLAIRE), la description textuelle concerne souvent des informations périphériques (comme son titre ou son état), alors que la photo présente des caractéristiques plus facilement généralisables pour un modèle image (format rectangulaire, titre généralement en haut au centre, etc.). L'interprétation avec Grad-CAM a montré que le modèle se concentre bien sur la localisation du texte dans l'image. À l'inverse, la richesse descriptive du texte pour décrire les articles de bricolage très variés rend les généralisations prédictives du modèle texte plus productives dans le cas de ces catégories.

En ce qui concerne la fusion précoce, le déséquilibre entre les étiquettes du jeu de données ainsi que l'hétérogénéité descriptive au sein d'une même étiquette entre contenus texte et image rendent les bénéfices prédictifs plus incertains. Cette architecture peut être moins robuste et susceptible d'introduire davantage de biais. Cependant, il semble que le modèle ait réussi à découvrir des dimensions de généralisation supplémentaires lors de l'apprentissage des caractéristiques combinées des entrées multimodales. Cela suggère qu'en dépit de ses limitations potentielles, la fusion précoce peut aussi offrir la possibilité au modèle d'appréhender les interactions entre modalités dès les premières étapes du processus d'apprentissage, même dans le cadre de données déséquilibrées et hétérogènes.

Nous aspirons à réaliser une analyse plus approfondie, notamment par étiquette, et à établir des mesures comparatives consolidées entre ces trois modèles de fusion multimodale. L'objectif était de déterminer le modèle le plus approprié pour le projet Rakuten. Toutefois, la fusion précoce a requis des ressources de calcul substantielles qui n'étaient pas à notre disposition ou qui ont été mobilisées au détriment de cette analyse détaillée. Cette contrainte a entravé notre capacité à mener une évaluation plus exhaustive de ces modèles.

## VI. Cr ation d'une API

Afin de finaliser le projet, nous d cidons de cr er une API robuste en utilisant FastAPI, un framework Python moderne, avec une connexion   une interface utilisateur interactive cr  e e avec Streamlit.

L'id e e globale est qu'un utilisateur remplisse un champ 'd esignation' et un champ 'description' relatif   l'objet qu'il souhaite mettre en vente. Il devra ensuite ajouter une image de l'objet correspondant. Voici un aper u:

The screenshot shows a Streamlit application with a dark theme. At the top is the Rakuten logo. Below it, the text "Bienvenue sur le site Rakuten" and "Que souhaitez-vous vendre aujourd'hui ?". A subtext encourages users to provide a precise title and category. The form contains fields for "D esignation \*", "Description", and a file upload section for "Chargez l'image de votre objet". The file uploaded is "marteau.jpeg" (3.6KB). A "Envoyer" button is at the bottom.

Rakuten

Bienvenue sur le site Rakuten

Que souhaitez-vous vendre aujourd'hui ?

Un titre pr  cis et la bonne cat  gorie, c'est le meilleur moyen pour que vos futurs acheteurs voient votre annonce !

D esignation \*

MARTEAU DE CHARPENTIER / MOBISTE

Description

Pour les m  tiers de la charpente et la construction de maison   ossature bois. Manche Nanovib<sup>®</sup> : a

Chargez l'image de votre objet

Drag and drop file here  
Limit 200MB per file • PNG, JPEG, JPG

marteau.jpeg 3.6KB

Envoyer

---

En cliquant sur le bouton “Envoyer”, une requête contenant les données est envoyée à FastApi, qui après analyse des données reçues et traitement, retourne sous forme de dictionnaire les éléments souhaités à savoir les 3 meilleures classes, leurs désignations textuelles ainsi que leurs probabilités. Voici comment l’utilisateur reçoit ces informations:



---

## VII. Conclusion

En conclusion, le challenge Rakuten nous a donné l'opportunité de pratiquer de façon transverse diverses compétences acquises lors de notre formation en data science : clustering, classification supervisée, interprétabilité, Deep Learning. Davantage encore, la problématique professionnelle et actuelle dans laquelle il s'inscrit nous a conduits à explorer des techniques avancées dans les domaines du NLP et de la Computer Vision.

Pour généraliser, nous avons observé que si l'assouplissement de leurs contraintes d'utilisation permet aux entreprises du secteur e-commerce d'accroître leur couverture, elles s'exposent ce faisant à une dégradation de la qualité des données. En mettant en place un modèle multimodal qui surperforme les modèles unimodaux, nous avons montré que le recours simultané à différentes natures de données permet de réduire ce risque.

Aussi, il nous a tenu à cœur de conserver la visée concrète du projet en inscrivant nos travaux dans le développement d'une solution utilisateur qui offre une amélioration objective pour l'utilisateur du site Rakuten et enrichit son expérience de vente.

Un aspect crucial de ce projet a été le travail collaboratif. La complexité et l'ampleur du projet ont rendu indispensable la collaboration entre les membres de l'équipe. La répartition du travail s'est naturellement faite en fonction de l'investissement de chacun et de ses intérêts propres, ce qui a non seulement favorisé une bonne entente au sein de l'équipe, mais a également permis une exploration plus approfondie des différentes facettes du sujet.

Nous tenons à remercier toute l'équipe DataScientest et en particulier notre mentor de projet Francesco Madrisotti dont les conseils ont su nous orienter tout au long de la réalisation de ce projet fil rouge.