Projet6 Classifiez automatiquement des biens de consommation

Analyse des données textes et images

AL SAMMAN Wassim *Data Scientist Apprenti*

PAPOUTSIS Panayotis *Data Scientist -Mentor*

Introduction et problématique

En tant que Data Scientist à l'entreprise Place du marché

- Automatiser la mission recherche des produits pour trouver sa catégorie

- Etudier la faisabilité d'un moteur de classification :
  - Réaliser un prétraitement des descriptions des produits et des images
  - Réduire la dimension
  - Réaliser un clustering

Présentation du jeu des données

Exemples des données textes

```
data['product_name'][94]
```

```
'BeYOUtiful Copper Repouss�� - Man With Dhol Showpiece  -  36 cm'
```

```
data['description'][94]
```

```
'Buy BeYOUtiful Copper Repouss�� - Man With Dhol Showpiece  -  36 cm for Rs.999 online. BeYOUtiful Copper Repouss�� - Ma
n With Dhol Showpiece  -  36 cm at best prices with FREE shipping & cash on delivery. Only Genuine Products. 30 Day Replacem
ent Guarantee.'
```

```
data['product_category_tree'][94]
```

```
'["Home Decor & Festive Needs >> Showpieces >> BeYOUtiful Showpieces"]'
```

Traitement du texte

Nettoyer le texte

- Supprimer les ponctuations et stop_words
- Transformer la phrase vers des mots avec word_tokenize
- Vérifier le longueur du mot et appliquer le cas lower
- Prendre la base des mots avec lemmatizer

Exemple

```
sentence = data['description'][500]
sentence
```

```
2]: 'Nexus NX_7668 Analog Watch  - For Men - Buy Nexus NX_7668 An
t Flipkart.com. - Great Discounts, Only Genuine Products, 30
```

```
clean_text(sentence)
```

```
3]: ['7668',
 'analog',
 'watch',
 'for',
 'men',
```

Traitement du texte

Bag-of-words et créer les features

- Modèle countvectorizer

```
corpus = data['product_name']
get_feature_countvectorizer(corpus)
```

	001	005	006	008	011	01433cmgy	01
0	0	0	0	0	0	0	
1	0	0	0	0	0	0	
2	0	0	0	0	0	0	

- Modèle tfidvectorizer

```
corpus = data['description']
get_feature_tfidvectorizer(corpus)
```

	000	001	0021	004	005	006	008
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0

```
corpus = data['product_name'] + data['description']
get_feature_countvectorizer(corpus)
```

	000	001	0021	004	005	006	008	0083	011	01433cmgy
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0

Traitement du texte

Réduire les composantes avec ACP

- La variance expliquée est plus de 80%

```
features = get_feature_tfidvectorizer(corpus=data['product_name']+data['description'])
```

```
features.shape
```

```
(1050, 5814)
```

```
features_pca = transform_features_pca(features)
```

```
features_pca.shape
```

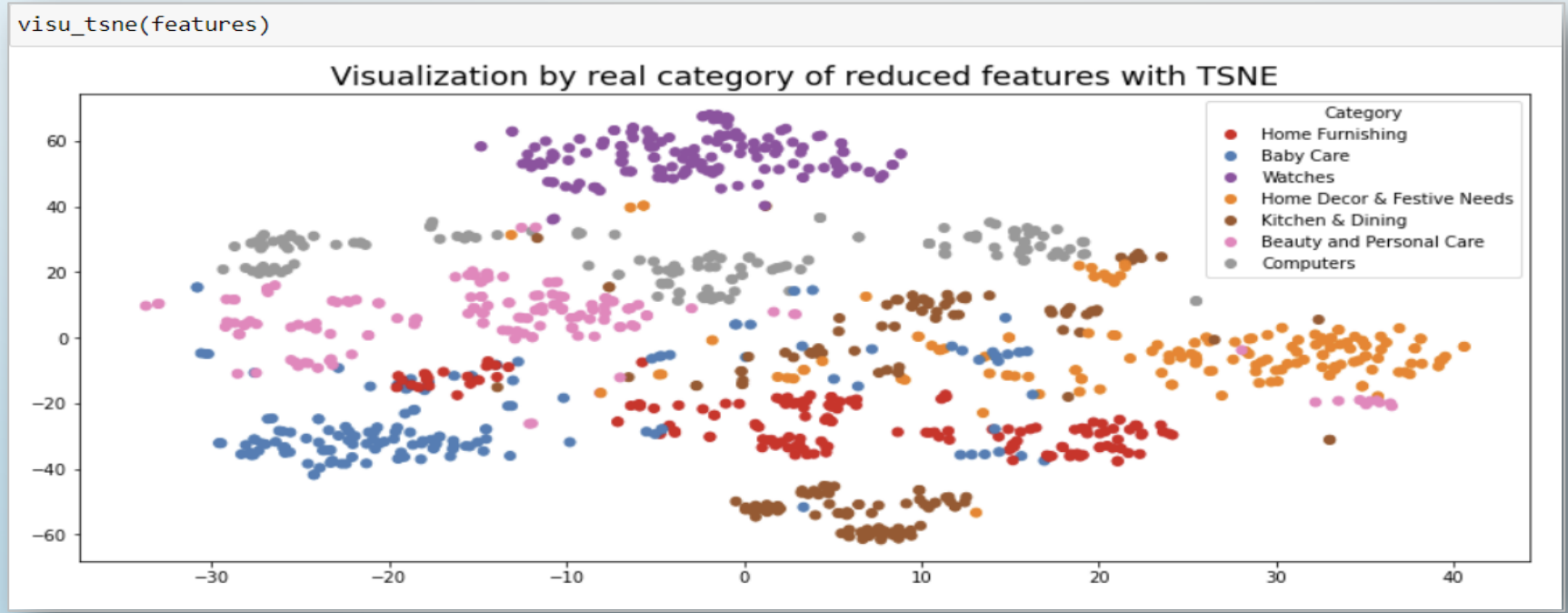
```
(1050, 510)
```

Traitement du texte

Réduire les composantes à 2 avec TSNE et présenter par catégorie réelle

- Préparer les catégories réelles
- Représenter les catégories par chiffres

```
visu_tsne(features)
```



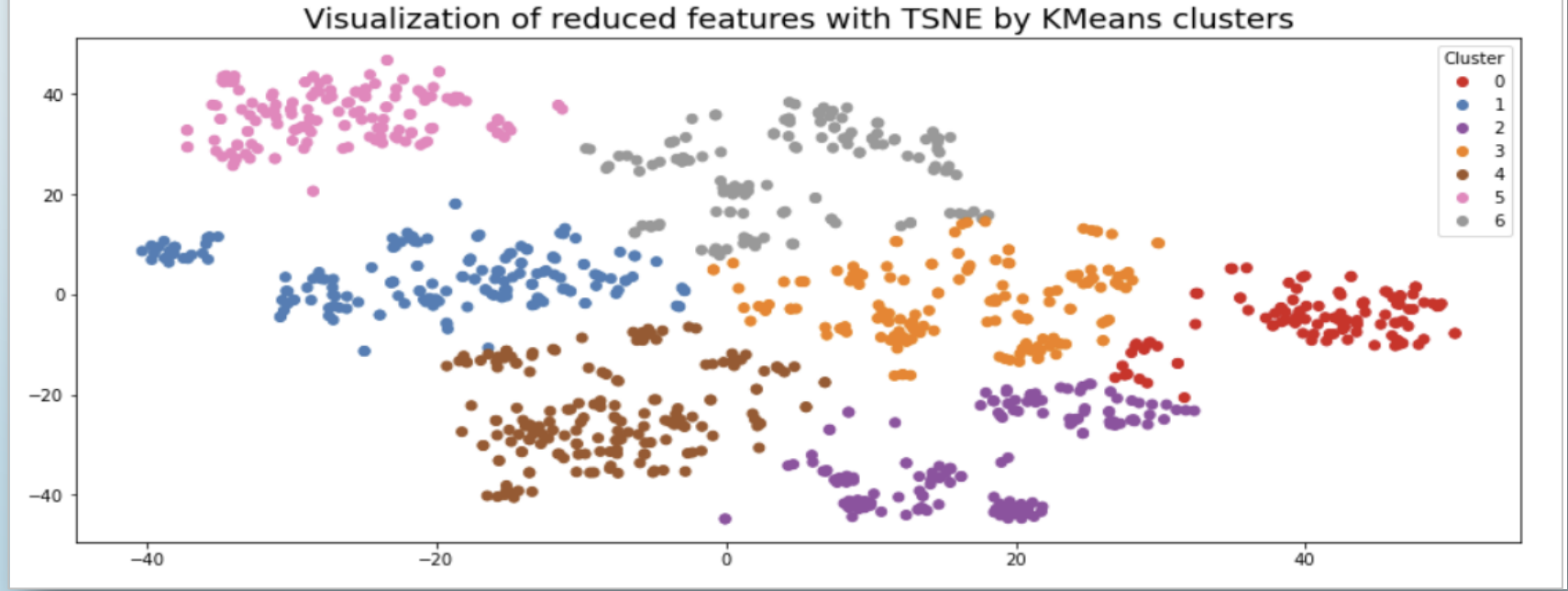
Traitement du texte

Réduire les composantes à 2 avec TSNE et présenter par cluster

- Ajouter un modèle KMeans

```
visu_tsne_kmeans(features);
```

```
ARIScore(Category, Clusters) = 0.44
```



Traitement du texte plus avancé

Comparer les modèles par catégorie réelle				
Modèle	Regroupement des données	La distance entre les groupes	Paramètres	
Word2Vec	Très dispersées	-	w2v_window = 5	w2v_min_count = 1
			w2v_epochs = 100	w2v_size = len(sentences)
BERT HuggingFace (Uncased)	Pareil	Pareil	max_length = 64	batch_size = 10
			model_type = 'bert-base-uncased'	
BERT HuggingFace (Cardiffnlp)	Mauvais regroupement	Pareil	max_length = 64	batch_size = 10
			model_type = 'cardiffnlp/twitter-roberta-base-sentiment'	
BERT hub Tensorflow	Pareil	Pareil	max_length = 64	batch_size = 10
			model_type = 'bert-base-uncased'	
USE-Universal Sentence Encoder	Pareil	Pareil	batch_size = 10	

Traitement du texte plus avancé

Comparer les modèles par cluster					
Modèle	Regroupement des données	La distance entre les groupes	ARI Score	Paramètres	
Word2Vec	Mieux regroupées	Très bien diminuée	Faible valeur 0.001	w2v_window = 5	w2v_min_count = 1
				w2v_epochs = 100	w2v_size = len(sentences)
BERT HuggingFace (Uncased)	Pareil	Pareil	Pareil	max_length = 64	batch_size = 10
				model_type = 'bert-base-uncased'	
BERT HuggingFace (Cardiffnlp)	Pareil	Pareil	Valeur moyenne 0.19	max_length = 64	batch_size = 10
				model_type = 'cardiffnlp/twitter-roberta-base-sentiment'	
BERT hub Tensorflow	Pareil	Pareil	Pareil	max_length = 64	batch_size = 10
				model_type = 'bert-base-uncased'	
USE-Universal Sentence Encoder	Pareil	Pareil	Pareil	batch_size = 10	

Traitement des images _ SIFT

- Préparer les features images

```
df_features_images = get_feature_sift()
df_features_images
```

- Réduire les features avec ACP

```
features = transform_features_pca(df_features_images.drop(columns=['real_categories']))
```

```
features.shape
```

```
(1049, 10)
```

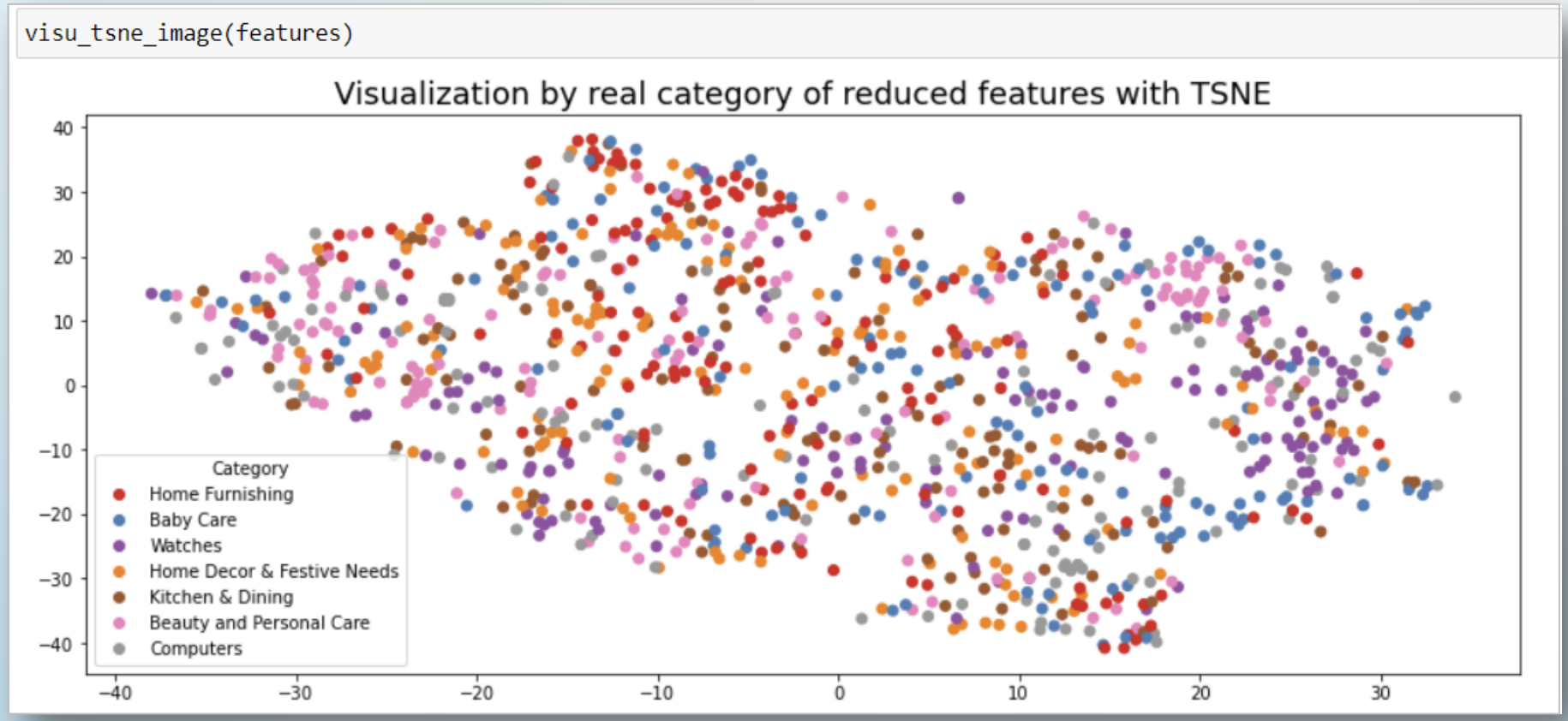
```
df_features_images[[0, 'real_categories']]
```

	0	real_categories
0	31.991209	Home Furnishing
1	31.096561	Baby Care
2	28.577772	Baby Care
3	28.321424	Home Furnishing
4	23.492832	Home Furnishing
...	...	...
1044	22.710911	Baby Care
1045	19.600000	Baby Care
1046	18.276722	Baby Care
1047	15.890792	Baby Care
1048	18.669973	Baby Care

1049 rows × 2 columns

Traitement des images _ SIFT

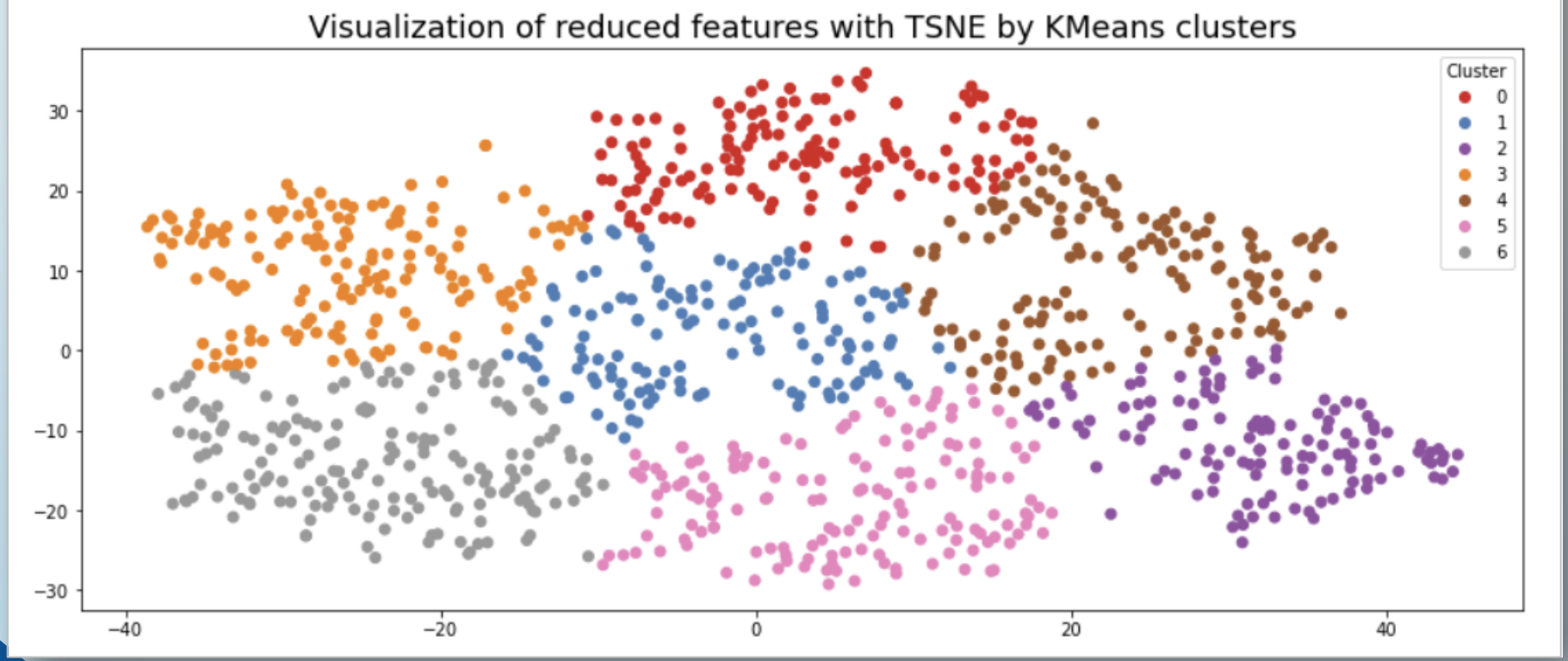
```
visu_tsne_image(features)
```



Traitement des images _ SIFT

```
visu_tsne_kmeans_image(features);
```

```
ARIScore(Category, Clusters) = 0.028
```



Traitement des images _ Transfer learning

- Préparer les features images

```
df_features_images = get_feature_vgg16()
```

- Réduire les features avec ACP

```
features = transform_features_pca(df_features_images.drop(columns=['real_categories']))
```

```
features.shape
```

```
(1050, 10)
```

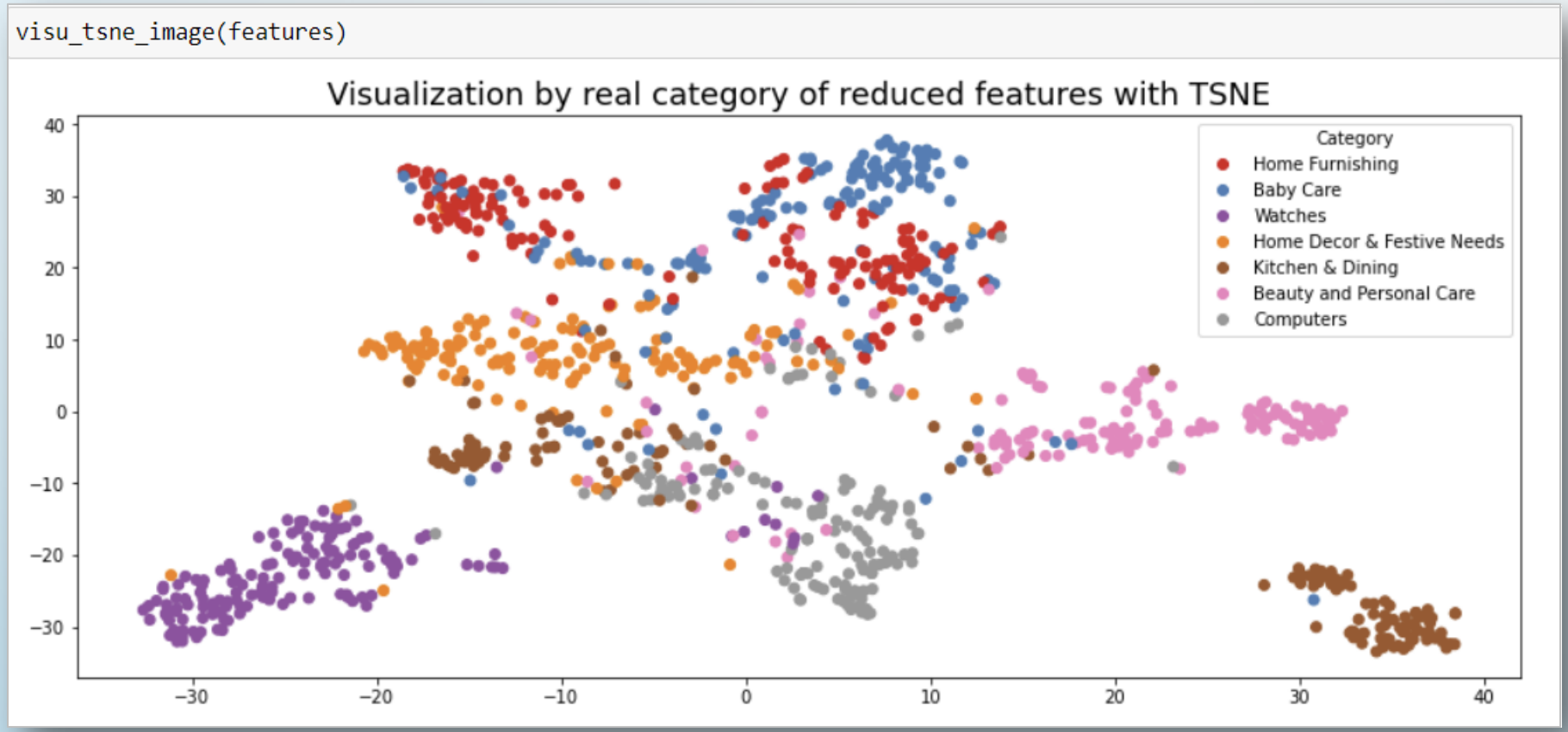
```
df_features_images[[0, 'real_categories']]
```

	0	real_categories
0	3.398615	Home Furnishing
0	0.000000	Baby Care
0	0.000000	Baby Care
0	0.000000	Home Furnishing
0	3.417152	Home Furnishing
...	...	...
0	2.312268	Baby Care
0	1.217184	Baby Care
0	2.196130	Baby Care
0	0.000000	Baby Care
0	0.000000	Baby Care

1050 rows × 2 columns

Traitement des images _ Transfer learning

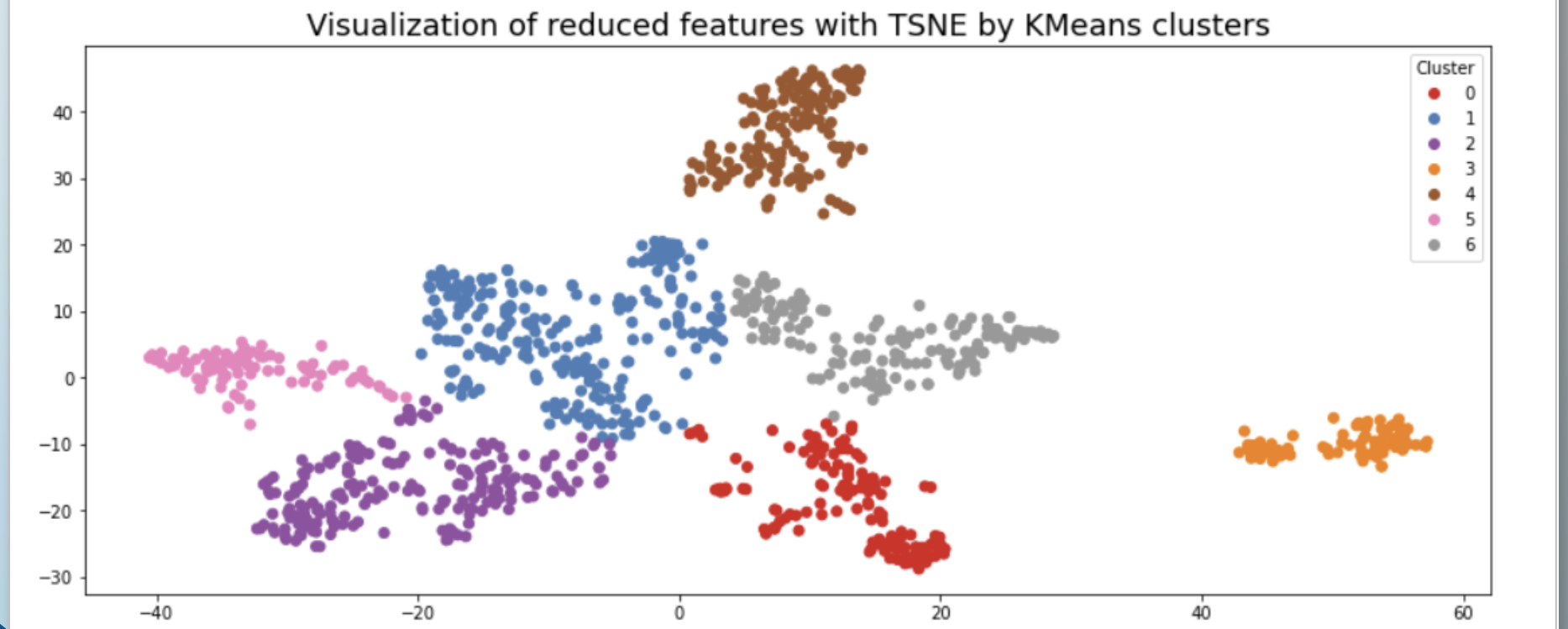
```
visu_tsne_image(features)
```



Traitement des images _ Transfer learning

```
visu_tsne_kmeans_image(features);
```

```
ARIScore(Category, Clusters) = 0.506
```



Traitement des images _ Comparaison

Comparer les modèles SIFT et VGG16			
Modèle	Regroupement des données	La distance entre les groupes	ARI Score
SIFT	Très dispersées	Petite distance	Faible score 0.028
Transfer learning VGG16	Bien regroupées	Distance moyenne	Très bon score 0.506

Conclusion générale

- Des nouvelles compétences acquises
- L'idée de réduire les composantes ACP ou TSNE
- Le modèle Word2Vec est non validé
- Le modèle VGG16 est meilleur que le SIFT
- Mon sentiment après le projet 6

Merci pour votre attention