11 March 2019

Alexandre ALLANI
Adrien CLOTTEAU

Rapport projet S5

Challenge Total

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Contents

1 Introduction

Vous trouverez l'ensemble de ce qui a été fait en suivant le lien ci-dessous.
`https://github.com/Syndorik/ChallengeTotal`.

En suivant le lien Github, vous trouverez:

- Challenge-Total-Overlook-of-the-data.ipynb : Jupyter Notebook sur la visualisation des donnés
- Data Engineering and Modeling.ipynb : Jupyter Notebook sur la préparation des données et la modélisation
- Time_series.ipynb : Jupyter Notebook sur la modelisation avec les Time Series.

2 Contexte

3 Visualisation des données

Le code lié à cette partie est disponible sur le lien suivant `https://github.com/Syndorik/ChallengeTotal/blob/master/Challenge-Total-Overlook-of-the-data.ipynb`.

4 Préparation des données (Data Engineering)

Le code lié à cette partie est disponible sur le lien suivant `https://github.com/Syndorik/ChallengeTotal/blob/master/Data-Engineering-and_Modeling.ipynb`.

Dans cette partie, nous allons présenter la préparation des données faites avant les dernières modélisation. En suivant la méthodologie CRISP, cette préparation des données est le fruit de plusieurs tests/modélisation. Nous ne présenterons que les résultats fructueux, qui ont mené à une amélioration du modèle.

Modification du jeu de données Le jeu de données étant Français et l'interpréteur utilisé étant sous le format anglo-saxon, les valeurs numériques (tels que la température, la nébulosité etc.) étaient considérés comme des chaînes de cratère à cause de la virgule. Nous avons donc remplacé les "," par des "." pour pallier à ce problème.
Par ailleurs, le jeu de donnée ne contenant aucune valeur non acquises (NA values), il n'y a pas eu de pré-traitement à faire.

4.1 Segmentation temporelle

4.1.1 Distinction des vacances

L'une des premières améliorations effectuée au niveau des données est la segmentation du jeu de données en période de vacances et période de "travail". En effet, il est important de modéliser cette saisonnalité car les ventes sont très différentes pendant et hors des périodes de vacances. Le graphique ci-dessus, **Figure 1**

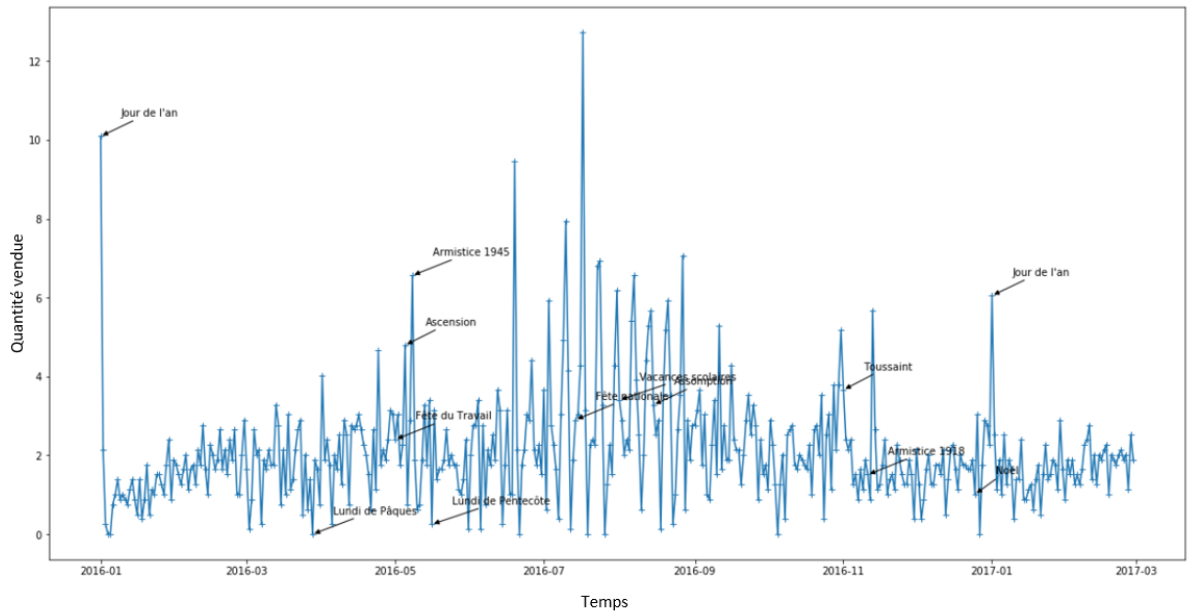

Figure 1: Vente de baguette au cours de la période

représente les ventes des baguettes sur les deux stations durant la période couverte par le jeu de données. Les baguettes sont le produit le plus vendus, tout produits confondus. Nous pouvons observer que les ventes lors des jours fériés et périodes de vacances sont beaucoup plus grandes. En particulier, les grandes vacances sont plus sujettes à de fortes variations.
Ces observations sont logiques, les périodes de vacances correspondes aux périodes de départs. Les familles vont en général plus s'arrêter dans les stations services, ainsi la quantité de produit vendu est beaucoup plus grande.

Segmenter l'année permet de mettre en évidence les périodes où les ventes sont très variables par rapport à la moyenne et donc permet d'affiner notre modèle.En se basant sur les **vacances 2015-2016** et les **vacances 2016-2017**, nous avons pu segmenter la période selon les vacances scolaires et les zones de départ en créant une nouvelle variable "holidays".

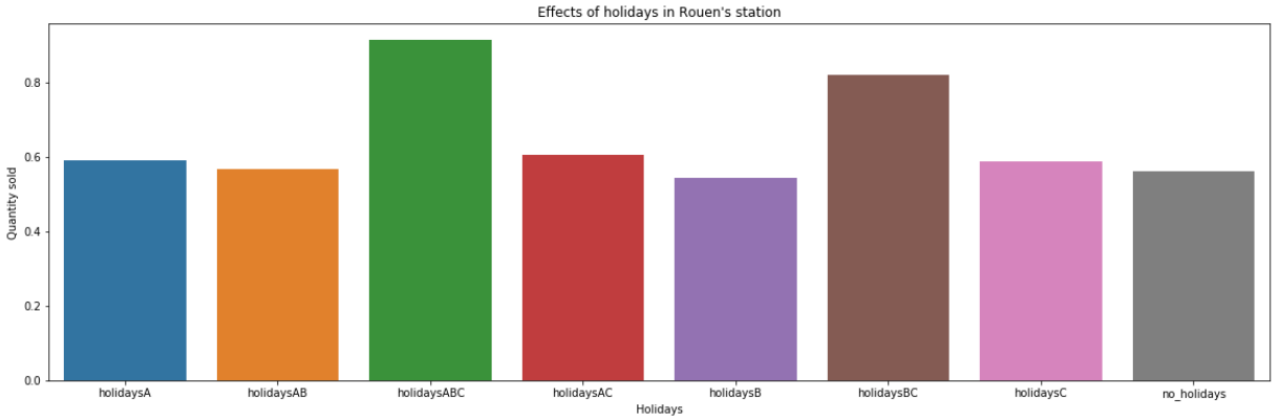

Figure 2: Vente cumulé par semaine selon la journée

Après création de ces variables, nous pouvons comparer les ventes moyennes entre les différentes périodes de vacances grâce à la **Figure 2**. Il apparait qu'il y a une réelle différence lorsque toutes les zones (A, B, C) sont en vacances ou juste les zones (B,C). Dans les autres cas, il n'y a pas beaucoup de différence avec les périodes de travails. Cela s'explique par le fait que les stations situés à Rouen sont en zone B, et la plupart des voyageurs se dirigeants vers Rouen doivent passer par Paris (zone C). Ainsi lorsque les deux zones sont en vacances en même temps, il y a bien plus d'aller-retour qu'en période normale. Nous choisissons de garder la discrimination entre les vacances pour les trois zones (A,B,C), les vacances pour les zones B et C, et finalement les périodes de "travail".

4.1.2 Distinction des dates de départs

Nous pouvons voir sur la **Figure 1** que les premières journées et dernières journées de vacances sont particulièrement haut en terme de quantité vendue. Cela n'est pas surprenant, ce sont des journées où beaucoup de famille sont sur les routes, ainsi il y a mécaniquement plus de ventes qui sont effectuées.
Il est donc intéressant de discriminer ces journées afin de mieux les modéliser. Pour cela nous avons créer une nouvelle variable "departure" qui segmente le jeu de donnée selon les dates de départs.

4.1.3 Segmentation de la période selon les journées et mois

Dans l'optique d'exploiter au maximum les saisonnalités des ventes, nous avons continué à travailler sur les dates. Comme nous pouvons le remarquer sur la **Figure 1**, les ventes atteignent un maximum local de manière périodique environ tout les 7 jours. Il y a une périodicité remarquable au niveau de la semaine. En effet, le trajet Paris Rouen ne dure que 2 heures, il est envisageable que certaines personnes font l'aller-retour chaque semaine pour rentrer le week-end, et donc s'arrêtent aux deux stations services.

La **Figure 3** représente le nombre de vente cumulé sur la période du jeu de donnée en fonction des jours de la semaine. Elle met en évidence cette périodicité. Les ventes des journées du Vendredi et Samedi sont bien au dessus des autres jours de la semaine. De manière générale, les ventes sont faibles pendant la semaine et haute pendant le week-end. C'est pourquoi nous avons différencié chaque jour de la semaine dans notre jeu de données.

4.1.4 Suppression de la colonne date

Une fois ces transformations réalisés, nous supprimons la colonne date. En effet pour les modèles que nous allons utilisé (excepté pour les Time Series), garder les dates va entraîner du sur-apprentissage. Les modèles

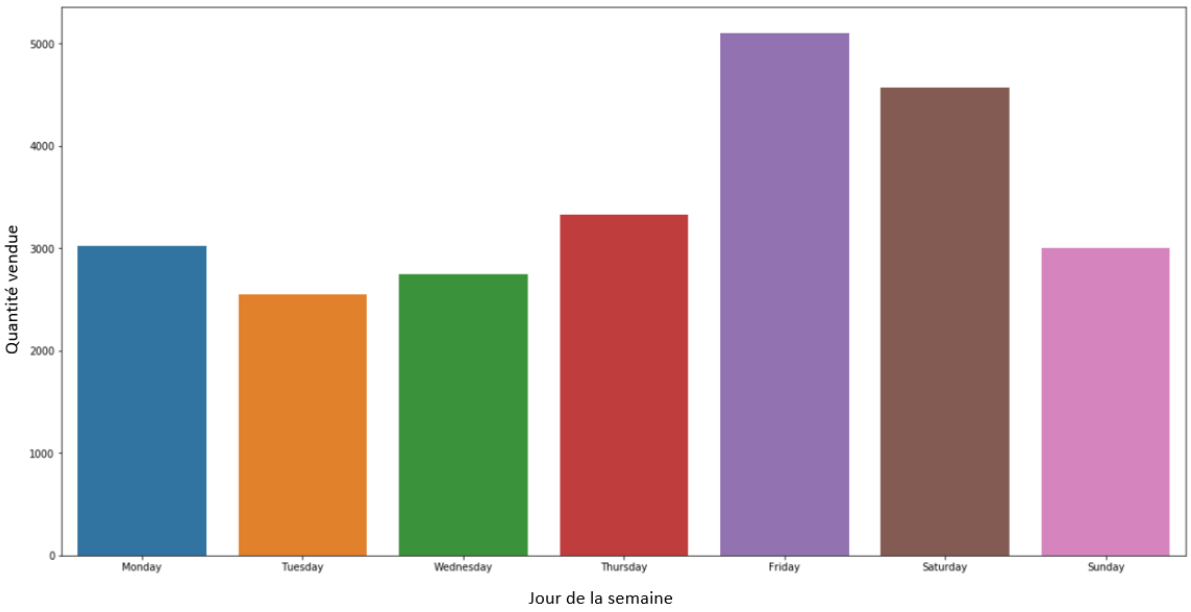

Figure 3: Vente cumulé par semaine selon la journée

vont apprendre sur la date exacte (jour-mois-année), sachant que ces dates sont passés, cela va entraîner une grande erreur.
Il est à noter que toutes ces segmentations temporelles ont pour but d'affiner le modèle, et retirer le plus d'information de la variable *date*. Cependant pour le modèle basé sur les Time Series l'ensemble de ces segmentations ne seront pas conservés, car incompatible avec le modèle. Nous expliciterons plus ce sujet dans la **section 5.2**

4.2 Influence des catégories

Nous voulions savoir si la distinctions par catégorie était utile, sachant qu'il y a inclusion des catégories : $cat6 \subset cat5 \subset cat4$. La **Figure ?? et ??** représente les ventes en fonction de la catégorie 5 pendant la période.

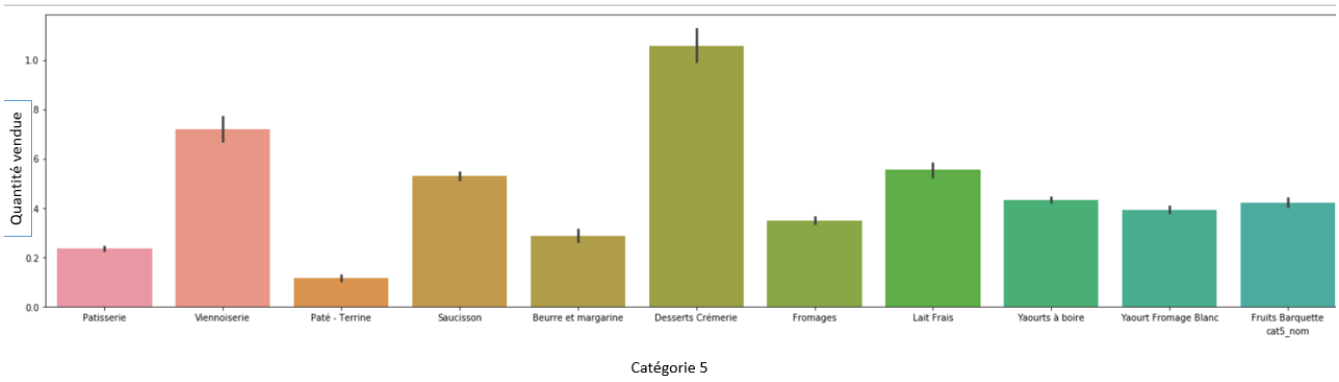

Figure 4: Vente cumulé en fonction de la catégorie 5 (partie 1)

Il est clair qu'il y a une différence de vente en fonction de la catégorie. La catégorie *Sandwich* ou encore *Dessert Crémerie* domine, tandis que la catégorie *pâté* et *autre charcuterie* font de faibles ventes. Cependant comme on peut le voir sur la **Figure 9** (Annexe 8.1), il y a peu de différence entre les sous catégories, en particulier entre la catégorie 5 et 6 qui paraissent être presque les mêmes. Cela s'explique par l'inclusion des catégories.
En réalisant les tests de modélisation, nous avons supprimé la catégorie 5 pour éviter la redondance des

données, ce qui n'a pas affecté nos modèles. Ainsi on conservait le regroupement par catégorie qui permet d'avoir une version plus grossière de nos produit, et un certain clustering, tout en limitant le nombre de variables pour éviter le *Fléau de la dimension* (*"curse of dimensionality"*).

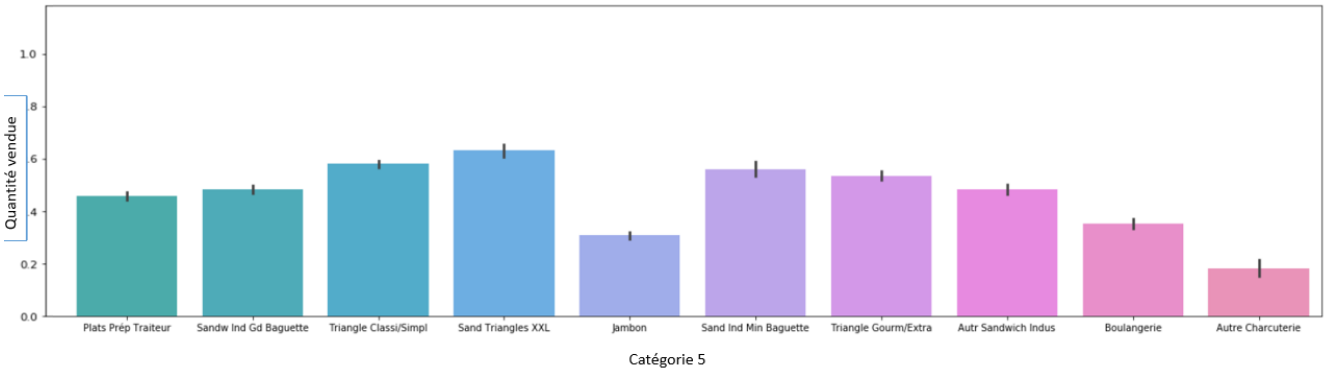

Figure 5: Vente cumulé en fonction de la catégorie 5 (partie 2)

4.3 Transformation des données

4.3.1 One Hot Encoding

Le One Hot Encoding est une transformation des données catégoriel en vecteur binaire. La **Figure 6** représente une telle transformation. Le principe est de créer un ensemble de nouvelles variables représentant

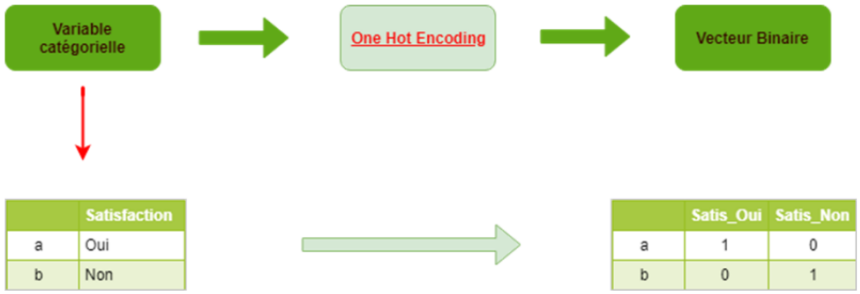

Figure 6: Principe du One Hot Encoding

une variables catégorielle (une variable par possibilité différente) prenant des valeurs binaires (0 ou 1). Il est important de faire cette transformation car plusieurs algorithme de machine learning n'arrivent pas à travailler avec les variables catégorielles. En l'occurrence, concernant notre projet les modèles suivants ne supportent pas les variables catégorielles:

- Modèle linéaire
- Réseau de Neurone

Cependant il y a plusieurs désavantage à cette technique:

- L'augmentation du nombre de dimensions qui est la cause de la *"curse of dimensionality"*. Le traitement de donnée devient plus difficile et long, car les données prennent plus de place en mémoire
- Les éléments du vecteurs sont équidistant, impliquant que chaque modalité d'une variable catégorielle ait la même importance ce qui n'est pas le cas. (Par exemple, le fait qu'il y a des vacances en zone A, B et C est plus important que le fait qu'il n'y ait pas de vacances)
- Le vecteur binaire ne contient une information sémantique encodée, ce qui rend plus difficile la compréhension.

Cette méthode étant simple à appliquer et permettant l'utilisation direct de la majorité des algorithme, nous allons tout de même l'utiliser dans la suite du projet.

4.3.2 Normalisation des données

5 Modélisation

5.1 Modélisation basée sur le Data Engineering

Le code lié à cette partie est disponible sur le lien suivant `https://github.com/Syndorik/ChallengeTotal/blob/master/Data-Engineering-and_Modeling.ipynb`.

5.2 Modélisation basée sur les Time Series

Le code lié à cette partie est disponible sur le lien suivant `https://github.com/Syndorik/ChallengeTotal/blob/master/Time_series.ipynb`.

6 Analyse des résultats

7 Conclusion

8 Annexe

8.1 Vente par Catégorie de produit

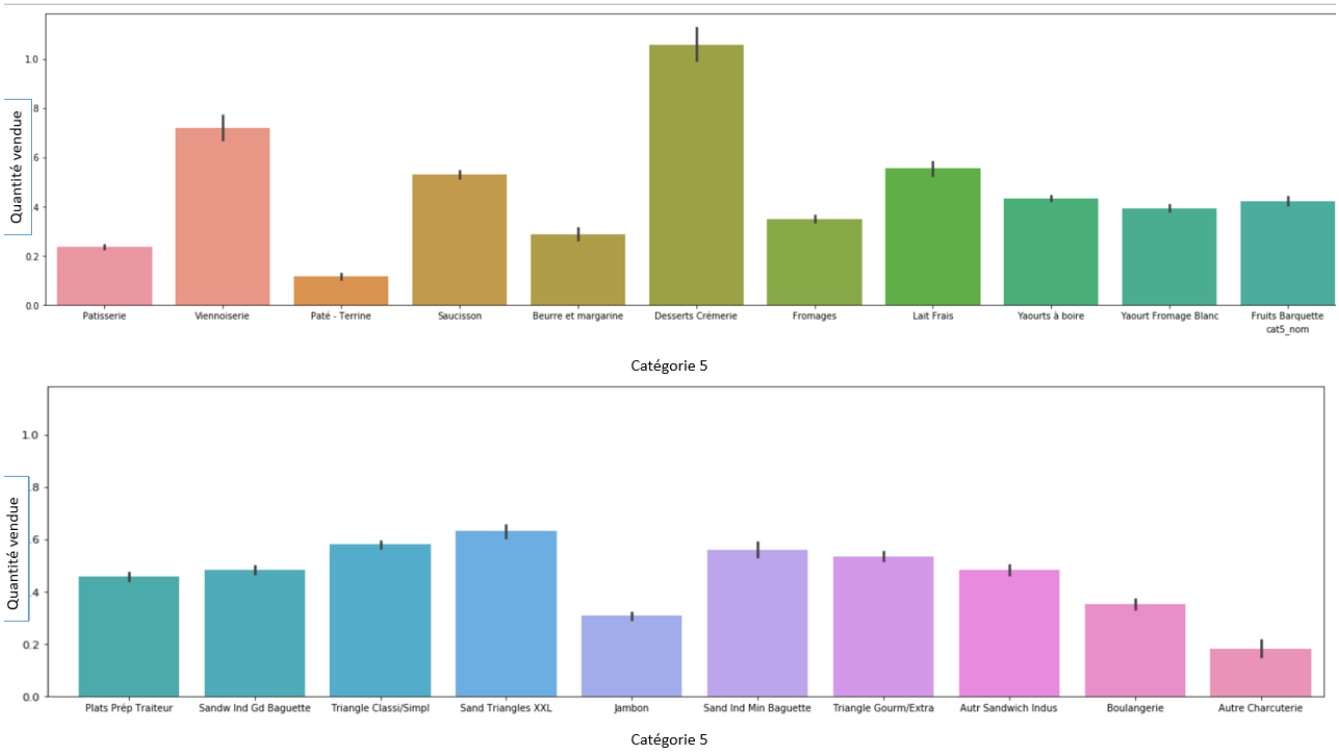

Figure 7: Vente cumulé en fonction de la catégorie 5

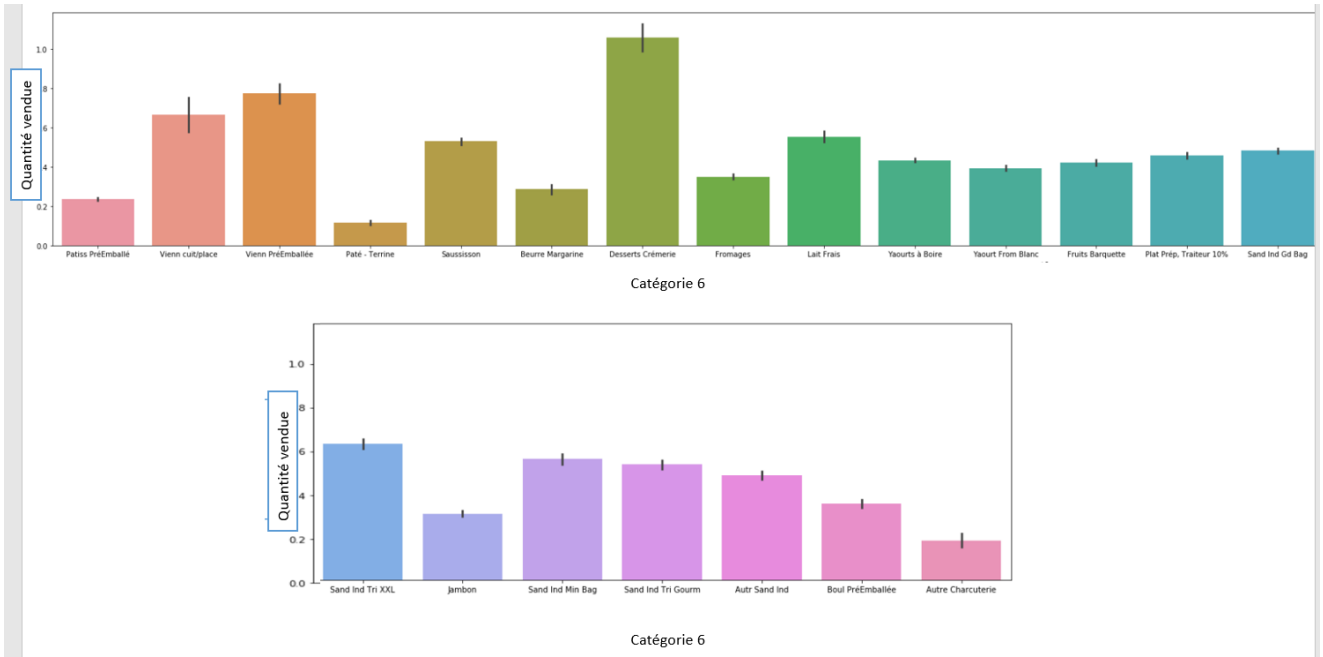

Figure 8: Vente cumulé en fonction de la catégorie 6

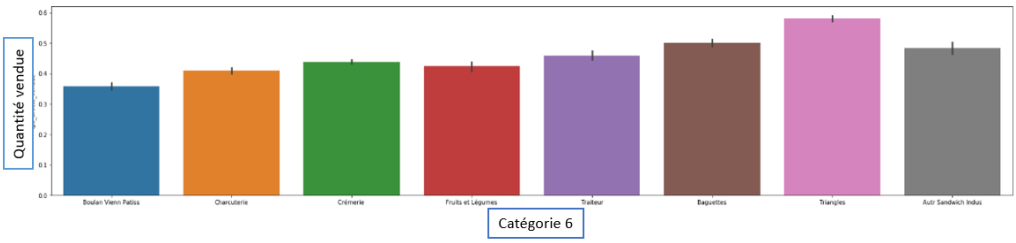

Figure 9: Vente cumulé en fonction de la catégorie 4