RAPPORT PROJET MICROSOFT AZUR MACHINE LEARNING

INTRODUCTION

Microsoft Azure Machine Learning Studio est un outil collaboratif fonctionnant par glisser-déplacer qui permet de générer, tester et déployer des solutions d'analyse prédictive à partir des données. Des algorithmes maisons, adossés à des méthodes reconnues, sont implémentées (régression logistique, forêts aléatoires, etc.). Nous pouvons démultiplier les analyses puisque Azure ML intègre le logiciel R et la grande majorité des packages associés. De fait, réaliser des traitements en ligne avec du code R est possible. Nous étudierons avec beaucoup de curiosité cette opportunité.

PREMIERE EXPERIENCE

PRESENTATION ET PREPARATION DES DONNEES

Dans ce jeu de données, chaque ligne représente un client et chaque colonne représente une variable associée au client. Nous allons prédire la variable **« Converted »** en fonction des autres variables associées à chaque client.

Pour pouvoir être analysé, un jeu de données nécessite généralement un traitement préalable. Dans un premier temps, nous remarquons les variables ContactDeptDesc, Nb_Catalog_Requests, Top_WebCat1, Top_WebCat2, Top_ProductFamily, Nb_checkout, Nb_Add_To_cart contiennent une grande quantité de valeurs manquantes. Nous allons donc les exclure dans la modélisation grace au module **Select columns in Dataset**.

Le module **Cleans Missing Data** a été ajouté deux fois pour remplacer les manquantes des variables qualitatives et quantitatives en utilisant respectivement le mode et la médiane. Ensuite les module **Remove Duplicate Row** et **Clips Values** pour supprimer les lignes en doubles et les valeurs aberrantes.

Après avoir nettoyés les données, le module **Feature Based Feature Selection** a été ajouté pour détecter le pouvoir prédictif entre la variable à expliquer et les variables explicatives.

On remarque certaines variables ont un pouvoir prédictif très faible (Source_organic, Type_visitor,Newsletter_Subscription,Nb_Add_To_cart,Account_Creation,InquireCount,Source_affiliate,Source_referral) par rapport à la variable dépendante et de ce fait on décide de les supprimer avec au module **Select columns in Dataset**. Ce module produit un jeu de données filtré qui contient uniquement les variables que nous souhaitons transmettre à l'algorithme d'apprentissage utilisé à l'étape suivante.

MODELISATION

Maintenant nos données sont prêtes, la construction d'un modèle de prévision passe par l'apprentissage et le test. Je vais utiliser les variables retenues pour former le modèle, puis tester le modèle pour voir dans quelle mesure il peut prédire la variable **Converted.**

La variable a prédire **Converted** prend 1 si la Journey s'est finie par une conversion 0 sinon. Ce qui nous permet de conclure, que cette variable ne peut pas suivre la loi binomiale et la modélisation de cette variable est un problème de classification binaire supervisée. Le modèle le plus adéquat pour ce genre de problématique est la régression logistique. Pour les besoins de nos données nous utilisons les modules **Two-Class Logistic Regression** pour l'apprentissage et **Two-Class Boosted Decision** pour le test.

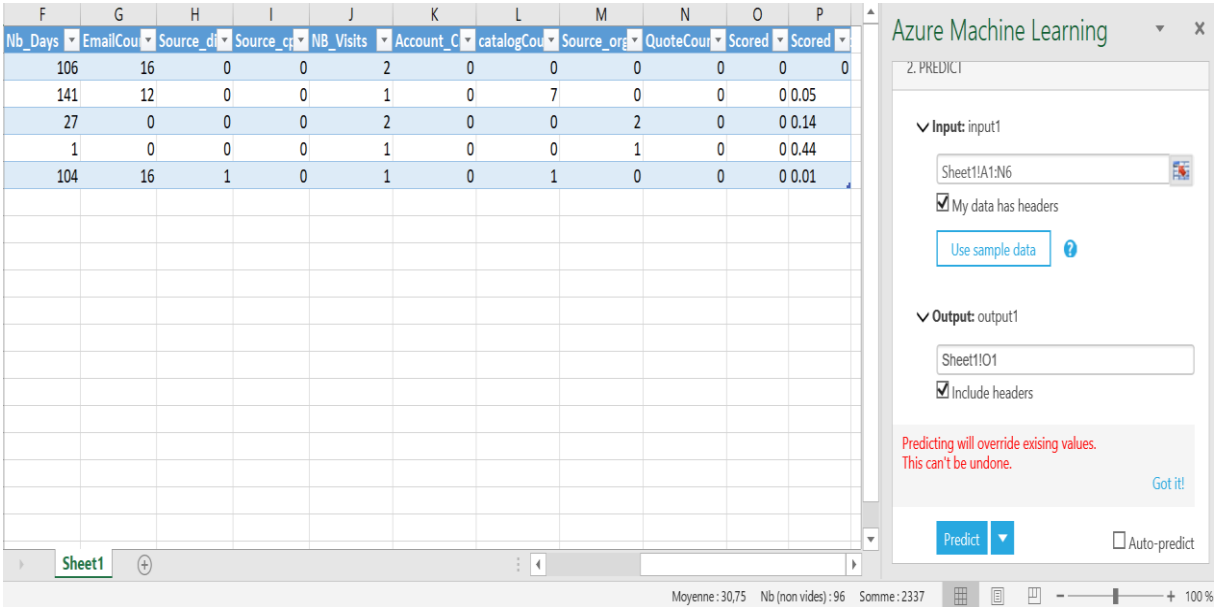
Pour former le modèle, je divise notre base de données en deux jeux de données distincts : 75% pour l'apprentissage et 25% pour le test. Ceci est fait grace au module **Split Data**.

À présent qu'on a formé le modèle à l'aide de 75 % de nos données, on peut l'utiliser pour la notation du reste de nos données (25 %), afin de voir s'il fonctionne correctement.

Les deux modules utilisés **Train Model** et **Evaluate Model** qui permettent d'entrer et évaluer les deux modèles.

D'après les mesures d'évaluation du classification binaire (Accuracy, Precision, Recall, F1 Score et AUC), le modèle de régression **Two-Class Boosted decision** est préférable contre le modèle de régression **Two-Class Logistic Regression** parcequ'il a les valeurs les plus élevées pour ces mesures. Ainsi ce modèle a été retenu et il sera utilisé pour le déploiement.

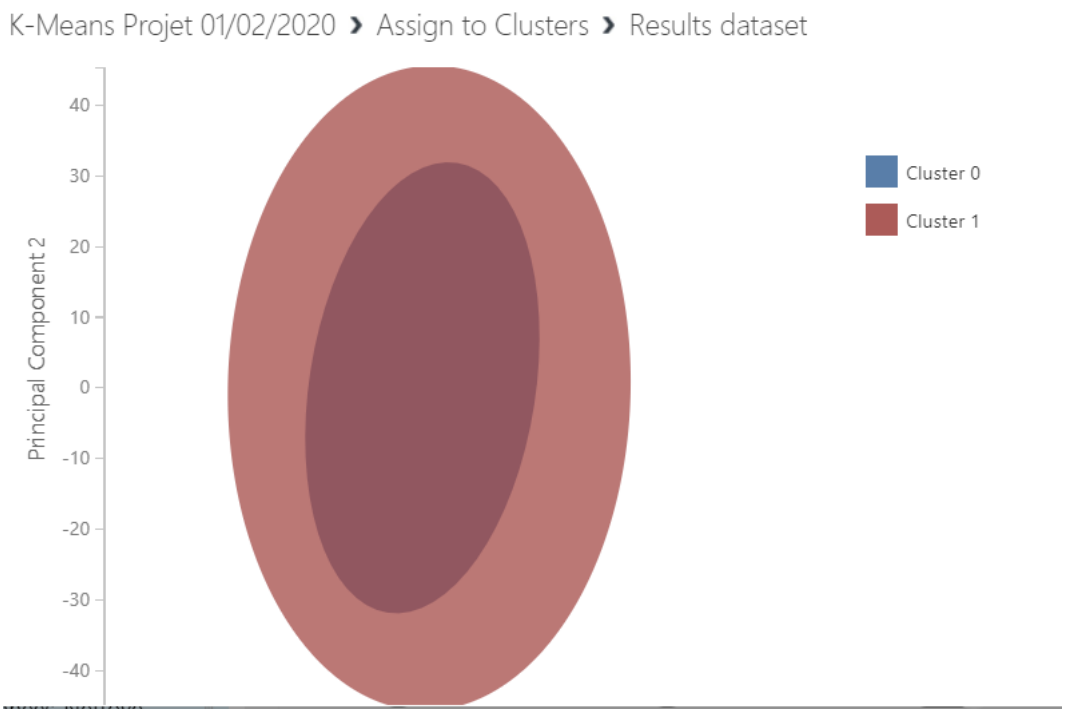
Le déploiement a été fait en premier lieu avec Excel et en second lieu avec python. Après un test de déploiement a été fait pour sortir les scores et les probabilités de ces deux types.



EXPERIENCE SUR LES K-MEANS (CLUSTERING)

L’algorithme K-means est un algorithme de Clustering. Ce dernier va mettre dans des zones (**Cluster**), les données qui se ressemble.

On ajoute la composante du modèle de clustering Train et les composantes du modèle de clustering K-means, puis on sélectionne des variables pour notre modèle d'algorithme k-means sur le modèle de cluster train. Et nous relie le module **Train Clustring Model** au module **Assign to Clusters.** Lorsque nous regardons les grappes de données assignées, cela crée deux grappes.



On a donc constaté que le nombre optimal de groupes pour notre ensemble de données est de deux, car la composante de regroupement par balayage teste certaines itérations et décide du paramètre optimal.