Rapport Knime

Mise en place des données sur Knime

Nous sommes partis d'un fichier au format CSV comprenant les 605 000 lignes fournies. En premier lieu, nous avons filtré les colonnes jugées "peu pertinentes". De ce fait, nous avons utilisés un column filter qui nous a enlevé les colonnes suivantes: id_product, code_barre, url, creator, created_datetime, last_modified_datetime, product_name, sub_category, completeness, last_image_datetime, image_url, image_small_url, nutrition_score_fr_100g, ecoscore_score, ingredients_tags, additives_tags, brands_tags.
Pour continuer dans la filtration des données, nous avons utilisé un Rule-Based Row Filter. Cela nous a permis premièrement d'éliminer les lignes avec des valeurs nulles. Deuxièmement, de vérifier que les valeurs nutritionnelles sur 100g soient bien comprises entre 0 et 100. Et enfin une ultime vérification de la conformité des nutri scores (de a à e). Pour faciliter la manipulation des données l'ecoscore est passé d'une notation en lettres (de a à e) à une notation en chiffres (de 1 à 5).

Exploration de données

Nous avons débuté par l'utilisation d'une matrice de corrélation pour déterminer quelles sont les données qui peuvent être intéressantes pour découvrir une potentielle synergie entre elles. Cela nous a permis de découvrir que l'énergie et le gras sont reliés entre elles. Mais aussi que le gras et les glucides sont négativement liés. Les protéines et les sucres sont eux aussi négativement liés. Mais le plus intéressant est que le nutri-scores est lié à la catégorie des produits.
On s'est servi d'un box plot pour visualiser les nombreux outliers. Cependant, les données n'étant pas sur les même échelle, certaines comprises entre 1 et 4 et d'autres entre 0 et plusieurs milliers. Nous avons utilisé un normalizer paramètres en normalisation min, max (entre 0 et 1) pour que les données soit plus facilement analysables sur les différents jeux de données. Cela a permis d'enlever une grande partie des outliers présents dans la base. Ainsi nous avons pu avoir des analyses d'avantages précises sur le futur.

Histogramme

Pour continuer l'exploration nous avons choisi d'utiliser un histogramme accompagné d'un color manager qui a permis de différencier par différentes couleurs les catégories de produits entre elles. Il a permis de décrire les différentes proportions de produits, ainsi que celles des outliers dans les différentes catégories. Donc on peut voir que la catégorie reine est celle des "sugary snacks". Et la catégorie Fat and sauces qui possède le plus grand pourcentage d'outliers par rapport à son jeu de données.

Clustering

Dans le cadre de la partie clustering du jeu de données nous avons utilisé les 3 principaux outils de clustering (k-Means, Hierarchical Clustering et le DBSCAN). Avant de les utiliser nous avons enlevé les outliers avec un Numeric Outilers permettant d'enlever automatiquement et de manière plus précise les outliers présents. De plus, l'avantage du Numeric Outliers est que les paramètres sont conservés quand nous relançons le Workflow. Dans le cas du Hierarchical Clustering et du DBSCAN nous nous sommes servis du Row Sampling. Nous avons sampler à 1% pour des raisons de puissances de traitement.
Et nous avons stratifié sur les nutri-scores dans le but de vouloir trouver un nouveaux modèles de scores.

k-Means

Nous avons utilisé k-Means pour des raisons de simplicité et d'efficacité. Celui-ci permet l'utilisation de l'intégralité du jeu de données sans avoir besoin de sampler. Après avoir utilisé des boucles pour tester les différents nombres de clusters (de 3 à 10 clusters car au-dessus de 10 cela n'a pas vraiment de pertinence de même pour la borne inférieur à 3). Nous avons pu déterminer que les meilleurs résultats étaient pour 6 clusters avec les paramètres suivants: fat_100g, saturated_fat_100g, sugar_100g, nova_group, ecoscore_grade. Pour mesurer la qualité de nos résultats, nous utilisons un entropy scorer mesurant donc l'entropie, la taille du cluster, l'entropy normalisé (l'entropie qui nous intéressent), ainsi que la qualité (elles aussi intéressante pour le clustering). Nous avons relevé une entropie normalisée de 0,732 avec une qualité de 0,268. Les résultats ne paraissent pas spécialement bons, mais ce sont les meilleurs résultats que l'algorithme peut nous proposer avec le jeu de données.

Hierarchical clustering

Pour repartir sur les mêmes bases nous gardons les colonnes précédemment utilisées ainsi que 6 clusters. De plus, nous ajoutons les paramètres de la fonction de distance en euclidienne et une mesure en moyenne.
Grâce à la fonction distance de l'algorithme, le choix de 6 clusters était assez pertinent. Nous avons mesuré l'entropie. Nous obtenons les résultats suivants , une entropie normalisée de 0,834, avec une qualité de 0,166. Le hierarchical clustering apporte des résultats moins bons. Cependant, cette fluctuation est possiblement liée au sampling de 1%.

DBSCAN

Nous procédons de la même manière que le hierarchical clustering à la différence que nous passons par un nœud numeric distance pour paramétrer le DbScan. Pour le nœud numeric distance, nous gardons une configuration euclidienne avec les mêmes colonnes sélectionnées. La configuration du DBSCAN se fait avec un epsilon de 0.7 et un minimum de points de 5. Nous obtenons en résultat une entropie normalisée de 0,916 et une qualité 0,084. Ce sont des résultats très médiocre par rapport aux autres méthodes de clustering.

Conclusion

Grâce à ces tests et résultats nous pouvons en conclure que l'algorithme K-means semble être le plus propice pour l'attribution des nouveaux scores aux différents produits.

Classification

Pour réaliser la classification, nous utiliser un arbre de décision avec un Decision tree learner et tree predictor.Après de nombreuses essais nous choisissons de paramétrer le Desicion tree learner de la manière suivante, nous paramétrons avec la colonne nutri score grade, en gini index et no pruning et un nombre minimum d'éléments de 20. Envoie cette méthode dans un tree predictor qui est paramétré à 70 000 lignes. Avec l'aide d'un scorer, nous pouvons voir que la moyenne du recall est de 0,7992, une précision moyenne de 0,802 et une accuracy de 0,811. Ce sont des résultats corrects pour le jeu de données. Pour tester le taux d'erreur moyen de cette fonction nous avons le choix d'utiliser une cross validation. Dans le meilleur des cas, nous sommes à un peu moins de 17% d'erreurs et dans le pire des cas, nous nous trouvons à un peu moins de 19% d'erreurs.

Pour tester la fiabilité, nous comparons la colonne nutri score et la prédiction dans un scorer.

Attribution du nouveau score

Le score que nous avons créé se base sur les colonnes de fat_100g, saturated_fat_100g, sugar_100g, nova_group, ecoscore. Ce dernier prend en compte les apports nutritifs des aliments, mais aussi le niveau de transformation et son impact écologique. Le but de ce score est dans un sens de créer des produits qui soit bon pour nous , tout en respectant l'environnement.
Nous avons mesuré l'attribution des produits sur ce nouveau score. Avec un arbre de décision composée d'un desicion tree learner paramétré en gini index, no pruning avec en class de référence le nouveau score. Quant au tree predictor, il est à 70 000 lignes pour respecter le jeu de données.
Nous avons appliqué cette méthode sur les différentes méthodes de clustering fait précédemment:

- k-Means on obtient un taux d'erreur de 0,175% et une accuracy de 99,825%
- Hierarchical Clustering on obtient un taux d'erreur de 0,409% et une accuracy de 99,591%
- DBSCAN on obtient un taux d'erreur de 0,647% et une accuracy de 99,353%

Nous pouvons donc confirmer que k-Means est l'algorithme le plus pertinent pour créer un nouveau score qui soit attribuable à tous les produits.