Romain Le Morvan

Applications des techniques d'apprentissage automatique et des données massives en microéconomie :

Prédiction de l'état de santé d'un individu de 60 ans et plus

ESG UQAM

ECO930J

2020

Table des matières

Introduction

Le projet a pour but de prédire l'état de santé d'un individu de 60 ans et plus. Pour réaliser cette étude, plusieurs techniques de machine learning vont être utilisées. Au cours de la dernière décennie, le machine learning a fait son apparition dans une multitude de domaine telle que, marketing (churn, scoring), bancaire (solvabilité d'un client), industrie (robot), automobile (voiture autonome).

Cette étude est consacrée au domaine de la santé. Les recherches sont en pleine essor. Une multitude d'entreprise spécialisée dans ce secteur ont vu jour. Leurs analyses portent sur l'imagerie médicale, permettant une meilleur détection des maladies. De plus le traitement personnalisé, prévision des épidémies, découverte et fabrication de médicaments. Le machine learning est en train de révolutionner toute l'industrie.

Nous allons nous intéresser aux caractéristiques des individus afin de prédire son état de santé, à partir de sa consommation d'alcool, de tabac, sédentarité, son éducation etc. En premier lieu nous allons préparer les données, puis réaliser les modèles et enfin étudier les techniques afin d'améliorer la précision.

Cette étude est réalisée avec python, c'est le langage le plus utilisé en data science.

Inventaire des données exploitables

Les bases de données sont des enquêtes sur la santé dans les collectivités canadiennes pour les années 2010, 2012 et 2014. Elles proviennent de odesi, un référentiel numérique pour les données en sciences sociales, incluant les données de scrutins depuis 2007.

Pour chaque années nous avons environ 65 000 observations et plus de 1000 variables.

La variable dépendante est *GEN_01*: Évaluation personnelle de la santé.

Elle se décompose en plusieurs catégories :

- Excellente
- Très bonne
- Bonne
- Passable
- Mauvaise

Le choix des variables explicatives est considérable, on distingue différents domaines, des variables :

- D'identification
- Géographique
- Démographique
- Blessures
- Choix alimentaires
- Activités
- Revenu
- Sommeil
- ….

Exploration et préparation des données

Tout d'abord j'ai fusionné les 3 bases de données de 2010,2012,2014 ; En gardant les variables communes, donc un total de 931 variables. Un total de 70061 observations, prenant en compte la population de 60 ans et plus.

j'ai dû choisir des variables pertinentes pour la prédiction de l'état de santé. Pour ce faire j'ai sélectionné des variables non corrélées à l'état de santé (exclusion des variables liées au diabète, cancer, hypertension, etc). De plus avec un taux de valeur manquante inférieur à 10%.

Ci-dessous les variables :

Variable dépendante :

GEN_01 : Evaluation personnelle de la santé (1 : Excellente ; 2 : Très bonne ; 3 : Bonne ; 4 : Passable ; 5 : Mauvaise)

Variable santé générale :

GEN_07 : Stress perçu dans la vie (1 : Pas du tout ; 2 : Pas tellement ; 3 : Un peu ; 4 :Assez ; 5 : Extrêmement)

Variables démographiques

DHHGAGE : Âge (12 : 60-64 ; 13 : 65-69 ; 14 : 70-74 ; 15 : 75-79 ; 16 : 80 et +)

DHH_SEX : Sexe (1 : Masculin ; 2 : Féminin)

DHHGMS : Etat matrimonial (1 : Marié(e) ; 2 : Union libre ; 3 : Veuf/Séparé/div ; 4 : Célibataire/jam)

DHHGHSZ : taille du ménage (1 : 1 pers ; 2 : 2 pers ; 3 : 3 pers ; 4 : 4 pers ; 5 : 5 pers et plus)

DHH_OWN : propriétaire ou locataire (1 : Oui ; 2 : Non)

SDCFIMM : immigrant (1 : Oui ; 2 : Non)

Groupe d'éducation
EDUDR04 : + haut niveau/étu. - rép. 4 niv (1 : < diplm.sec ; 2 : Dipl.sec ; 3 : Etude postsec ; 4 : Diplm post-sec)

Groupe Taille et poids - Autodéclarés
HWTGBMI : IMC / autodéclaration - (D,G) ou HWTGISW: Class. IMC (18 +) / autodéclar. - (D,G) (variable continu, sup à 999,96 => Nan)

Groupe Vaccin contre la grippe
FLU_160: A déjà reçu un vaccin contre la grippe (1 : Oui ; 2 : Non)

Groupe utilisation de santé
HCU_1AA : à un médecin régulier (1 : Oui ; 2 : Non)

Groupe Général

PACDPAI : indice de l'activité physique (1 : Actif 2 : Modérément actif 3 : Inactif)

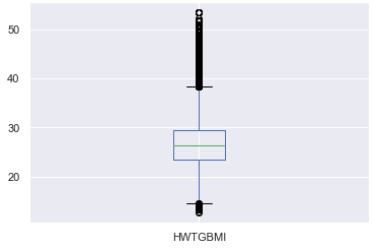
ALCDTTM : Genre de buveur (12 mois) (1 : buveur régulier ; 2 : buveur occasion ; 3 : pas bu 12 der. m)

SMK_01A : A fumé 100 cigarettes ou plus – vie (1 : Oui ; 2 : Non)

Traitement des valeurs manquantes

Le pourcentage de valeurs manquantes

GEN_01	0.29
DHHGAGE	0
DHH_SEX	0
DHHGMS	0.23
DHHGHSZ	0.03
DHH_OWN	3.2
SDCFIMM	3.51
FVCGTOT	11.77
EDUDR04	3.4
HWTGBMI	6.8
FLU_160	3.96
INCG2	12.69
HCU_1AA	0.05
ALCDTTM	2.32
PACDPAI	3.89
SMK_01A	0.39
GEN_07	1.01

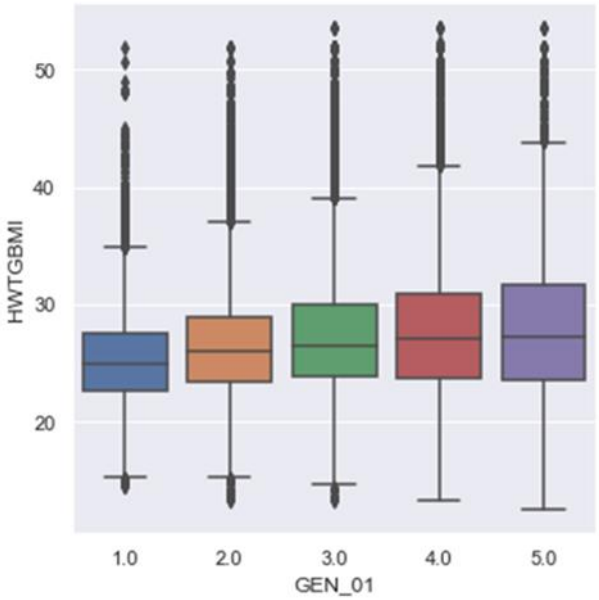


FVCGTOT (Cons. quoti. - tot. fruits/lég.) et *INCG2*(Rev. total/mén.) ont un pourcentage supérieur à 10%, donc je les supprime. On peut voir que d'autres variables ont un taux de valeurs manquantes non négligeable. On pourrait procéder par imputation, cependant on a pu voir à travers le boxplot que HWTGBMI à une forte variance. J'ai finalement enlevé toutes les observations avec au moins une valeur manquante. Le nombre d'observations passe de 70061 à 60923, soit une baisse de 13,04%.

Traitement des valeurs aberrantes

Il n'y a pas de valeurs aberrantes pour les variables catégoriques de l'étude, on ne peut pas observer de valeurs aberrantes. En effet les individus qui ne pratiquent pas d'activité sportive, les consommateurs réguliers d'alcool ou les fumeurs peuvent se sentir en bonne santé. En revanche L'IMC, la seule variable non catégorique, est un indicateur clé sur l'état de santé.

Ci-dessous la répartition de L'IMC des individus pour chaque catégories de l'état de santé :



IMC ≤ 18,5	Dénutrition
18,5 ≤ IMC < 25	Corpulence normale
25 ≤ IMC < 30	Surpoids
30 ≤ IMC < 35	Obésité modérée
35 ≤ IMC < 40	Obésité sévère
IMC ≥ 40	Obésité morbide ou massive

D'après le boxplot, on remarque des valeurs aberrantes. En effet les individus considérés comme en excellente ou très bonne santé présentent des IMC supérieur à 35 ou inférieur à 18.

Je décide donc de supprimer ces observations qui sont aberrantes.

Voici ma proposition pour chaque catégorie, je supprime :

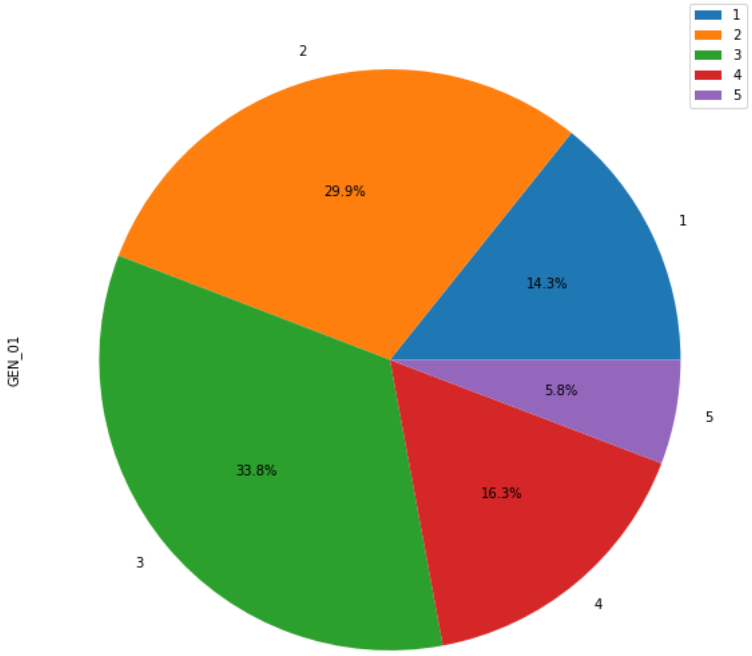
1- les observations où l'IMC est supérieur à 30 ou inférieur à 18,5
2- les observations où l'IMC est supérieur à 30 ou inférieur à 18,5
3- les observations où l'IMC est supérieur à 35 ou inférieur à 18,5
4- les observations où l'IMC est supérieur à 40 ou inférieur à 17

On dénombre 7225 observations

Statistiques descriptives

On peut voir que la suppressions des valeurs manquantes ou aberrantes n'a pas réellement modifié la répartition de la variable dépendante.

Echantillon	Excellente	Très bonne	Bonne	Passable	Mauvaise	Nbr observations
Initial	13,9%	31,7%	32,8%	15,8%	5,9%	70 061
Suppression des valeurs manquantes	14,4%	32,8%	32,6%	15,1%	5,1%	60923
Final	14,2%	29,8%	33,8%	16,3%	5,9%	53 698



Homme	Femme
43%	57%

60-64 ans	65-69 ans	70-74 ans	75-79 ans	80 et plus
26%	23%	18%	18%	15%

Actif (activité physique)	Modérement Actif	Inactif
23%	25%	52%

Buveur régulier	Buveur occasionnel	Pas bu les 12 derniers mois
55%	18%	17%

Corrélation des variables avec la variable dépendante :

Index	GEN_01
GEN_01	1
DHHGAGE	0.112085
DHH_SEX	-0.0240875
DHHGMS	0.0779968
DHHGHSZ	-0.0462102
DHH_OWN	0.140132
SDCFIMM	-0.00646942
EDUDR04	-0.175358
HWTGBMI	0.157249
FLU_160	-0.0802517
HCU_1AA	-0.0287532
ALCDTTM	0.194924
PACDPAI	0.256394
SMK_01A	-0.0813818
GEN_07	0.214282

(a)

Index	GEN_01
GEN_01	1
DHHGAGE	0.111439
DHH_SEX	-0.0271637
DHHGMS	0.0792489
DHHGHSZ	-0.0456458
DHH_OWN	0.146042
SDCFIMM	-0.00444831
EDUDR04	-0.18252
HWTGBMI	0.269988
FLU_160	-0.0839025
HCU_1AA	-0.029386
ALCDTTM	0.203925
PACDPAI	0.272321
SMK_01A	-0.0854762
GEN_07	0.221239

(b)

Le coefficient de corrélation de Pearson permet d'analyser les relations linéaires.

Le tableau (a) correspond à l'échantillon avant la suppression des valeurs aberrantes

Le tableau (b) correspond à l'échantillon final. L'activité physique à la plus forte corrélations avec 27,3%, vient ensuite L'IMC avec 26,9% (augmentation de 11% par rapport au tableau (a)) et l'état de stress avec 22,1%.

Pour les variables catégoriques, j'ai créé des variables dichotomiques. On a désormais 29 variables explicatives.

DHHGAGE	DHHGMS	DHHGHSZ	EDUDR04	PACDPAI	ALCDTTM	GEN_07
DHHGAGE_13	DHHGMS_2.0	DHHGHSZ_2.0	EDUDR04_2.0	PACDPAI_2.0	ALCDTTM_2.0	GEN_07_2.0
DHHGAGE_14	DHHGMS_3.0	DHHGHSZ_3.0	EDUDR04_3.0	PACDPAI_3.0	ALCDTTM_3.0	GEN_07_3.0
DHHGAGE_15	DHHGMS_4.0	DHHGHSZ_4.0	EDUDR04_4.0			GEN_07_4.0
DHHGAGE_16		DHHGHSZ_5.0				GEN_07_5.0

DHH_SEX	DHH_OWN	SDCFIMM	FLU_160	HCU_1AA	SMK_01A
DHH_SEX_1	DHH_OWN_1.0	SDCFIMM_1.0	FLU_160_1.0	HCU_1AA_1.0	SMK_01A_1.0

Elaboration des modèles

Dans cette section, je vais présenter différents modèles afin de prédire l'état de santé d'un individu.

J'utilise la Kfold cross validation, afin de séparer mon échantillon en deux catégories :
- apprentissage, qui permet d'apprendre sur le jeu de données

- test, afin de mesurer l'erreur d'affectation.

Cette méthode consiste à diviser les données en k sous-ensembles différents. k-1 ensemble(s) pour l'apprentissage et le dernier sous ensemble pour le test.

Modèles

Régression logistique multinomiale

La régression logistique multinomiale est une méthode de classification qui généralise la régression logistique à des problèmes multiclasses, c'est-à-dire avec plus de deux résultats discrets possibles, ici nous avons cinq catégories. L'objectif est de prédire les probabilités des différents résultats possibles de la variable dépendante, étant donné les variables explicatives.

Arbre binaire de décision

Une méthode de classification supervisée sous forme d'un arbre. A partir d'une racine que l'on divise de manière dichotomique en une séquence de nœuds. On choisit la variable explicatives qui est la « meilleur » division, et le seuil est déterminé selon qu'il minimise la somme des carrés des résidus.

Random Forest

Une amélioration du bagging pour les arbres de décision.

Gradient boosting

L'idée est de construire itérativement un modèle additif de prédiction en utilisant un arbre à chaque étape. Cette procédure équivaut à la minimisation d'une fonction de perte spécifique

Résultats des modèles

Les résultats présentent l'échantillon Final (1) et sans le traitement des valeurs aberrantes (2).

Précision des modèles, cross validation accuracy score (k=4, Train 75%) :

Echantillon	Logit Multinomial	Decision Tree	Random Forest	XGBclassifier	Gradient Boosting
(1)	39,66%	32,45%	36,36%	41,10%	41,95%
(2)	38,71%	29,72%	29,72%	38 ,11%	39,16%

Avec une autre méthode d'affectation Train/Test split

Précision des modèles, Train/Test split accuracy score (Train 75%) :

Echantillon	Logit Multinomial	Decision Tree	Random Forest	XGBclassifier	Gradient Boosting
(1)	39,66%	32,57%	36,48%	41,10%	41,95%
(2)	38,57%	29,92%	33,04%	38,41%	39,27%

Les deux types d'échantillonnage ont des résultats très proche.

Le traitement des valeurs aberrantes a permis d'améliorer les scores de précision malgré une perte de 7225 observations.
Le modèle Gradient Boosting présente la meilleur précision avec 41,95%.

	precision	recall	f1-score	support
1.0	0.38	0.09	0.15	1903
2.0	0.41	0.64	0.50	4013
3.0	0.42	0.54	0.48	4544
4.0	0.51	0.15	0.23	2171
5.0	0.51	0.15	0.24	794
accuracy			0.42	13425
macro avg	0.45	0.31	0.32	13425
weighted avg	0.43	0.42	0.38	13425

Précision

```
[[ 172 1273  443   13    2]
 [ 183 2550 1243   31    6]
 [  84 1855 2466  100   39]
 [  10  491 1280  321   69]
 [   6  101  398  166  123]]
```

Matrice de confusion

Le tableau de précision nous indique que les modalités 1,2 et 3 sont les plus difficile à prévoir.

La matrice de confusion permet de confronter les valeurs observées de la variable d'intérêt avec celles prédites par le modèle. On retrouve le taux de précision de 41,95%. On divise la somme de la diagonale de la matrice par le total (5632/13425).

Le taux de précision est de 41,95%, c'est un score assez faible. Par la suite nous allons essayer de comprendre ce résultat.

Hyperparamètre

Les hyperparamètres sont les différents paramètre d'un modèle qui permettent de diminuer la variance, d'éviter l'overfitting, ou bien d'améliorer la précision. Par exemple pour les arbres de décision, la profondeur des arbres, le nombre d'arbres et d'observations, etc.

La méthode Grid Search

C'est une méthode d'optimisation (hyperparameter optimization) qui va nous permettre de tester une série de paramètres et de comparer les performances pour en déduire le meilleur paramétrage.

L'état de santé de l'individu n'est pas mesuré par des tests ou analyses médicales. C'est le ressenti d'un individu sur son propre état de santé. Et l'on sait que chaque individu est unique et chacun à une perception différente sur son état de santé. D'après le tableau de précision, on remarque que les modalités 1,2 et 3 ont la plus faible précision. Il est difficile de différentier un état de santé entre les niveaux « excellent », « très bonne » et « bonne ». C'est pourquoi je propose une alternative pour prédire l'état de santé :

(1) Recoder la variable dépendante en 3 modalités.

- 5 : Mauvaise
- 4 : Passable
- 3 : Bonne

Désormais la catégorie 3 contient les états de santé « excellent », « très bonne » et « bonne ».

Remarque : Une autre alternative possible aurait pu être : Êtes-vous en mauvaise santé ?

- 1 : Oui
- 2 : Non

Avec la catégorie 2 contenant les états de santé « excellent », « très bonne » , « bonne » et « passable ». Cependant seulement 6,2% sont en mauvaise santé, donc peu d'intérêt à prédire cette alternative.

Résultats (accuracy score Train/test split 25%)

Logit Multinomial	Decision Tree	Random Forest	XGBclassifier	Gradient Boosting
78,44%	68,17%	76,47%	79,51%	79,93%

On constate une forte augmentation du score, que l'on peut estimer comme convenable.
Le gradient boosting à la meilleur précision avec 79,93%

Pour le Grid Search j'ai exécuté plusieurs paramétrages, mais je n'ai pas réussi à augmenter la précision. Soit le temps de calcul été trop lourd et j'ai dû arrêter l'exécution ou soit le score de précision n'a pas augmenté.

Conclusion

Cette étude avait pour but de prédire l'état de santé d'un individu de 60 ans et plus, à partir de variables non liées aux pathologies, maladies. Tout d'abord nous avons pu voir que le fait d'avoir 5 modalités pour la variable dépendante complique la prédiction. En effet obtenir des résultats précis sur l'état de santé, affaiblie la capacité de prédiction. Comme dit précédemment au niveau de la différenciation, il est difficile de choisir entre un état de santé « excellent », « très bonne » et « bonne ». Le choix est influencé par le point de vue de la personne, selon son opinion. Ce n'est pas mesuré ou testé par des analyses médicales. De plus les données de sondages présentent des inconvénients, comme la non vérification ou la présence de valeurs aberrantes, on a pu le voir avec L'IMC qui en dénombre énormément.
De plus le taux de variables catégoriques, elles représentent 96% des variables explicatives. En général, les variables numériques permettent une différenciation plus forte.
Enfin on peut conclure qu'il assez difficile de prédire l'état de santé à partir de variable non liées aux pathologies, maladies. Il aurait pu être intéressant d'avoir des variables selon l'alimentation, la pollution en ville et la durée de sommeil. Un des facteurs qui joue un rôle déterminant est la génétique, qui n'est pas réellement mesurable.

Malgré ces difficultés, nous pouvons retenir le Gradient boosting permettant de classer l'état de santé selon 3 modalités : « bonne », « passable » et « mauvaise ». Il présente une précision raisonnable de 80%.

Annexe

```
"""

Created on Wed Feb 12 18:14:50 2020


@author: Romai

"""


# importation des données

import pandas as pd

import scipy.stats

import numpy as np

df_2010 = pd.read_sas('C:/Users/Romai/OneDrive/Documents/UQAM/BD & ML/Projet micro/composante_annuelle_2010.sas7bdat',format='sas7bdat')

df_2012 = pd.read_sas('C:/Users/Romai/OneDrive/Documents/UQAM/BD & ML/Projet micro/composante_annuelle_2012.sas7bdat',format='sas7bdat')

df_2014 = pd.read_sas('C:/Users/Romai/OneDrive/Documents/UQAM/BD & ML/Projet micro/composante_annuelle_2014.sas7bdat',format='sas7bdat')


frames = [df_2010, df_2012, df_2014]


# fusion des bases de données

result = pd.concat(frames)

df = pd.concat(frames,join='inner')



# on sélectionne les individus qui ont 60 ans ou plus

df = df[df.DHHGAGE >= 12 ]



# on regarde le taux de valeurs manquantes des variables

df.FVCDFRU.isna().mean().round(4) * 100
```

```
# on sélectionne nos variables
df = df[['GEN_01','DHHGAGE','DHH_SEX', 'DHHGMS', 'DHHGHSZ', 'DHH_OWN', 'SDCFIMM',
        'FVCGTOT', 'EDUDR04', 'HWTGBMI', 'FLU_160', 'INCG2', 'HCU_1AA',
        'ALCDTTM', 'PACDPAI', 'SMK_01A','GEN_07']].copy()

# export dataframe to excel
df.to_excel(r'C:\Users\Romai\OneDrive\Documents\UQAM\BD & ML\Projet micro\dff.xlsx', index = False)

import pandas as pd
import numpy as np
df1 = pd.read_excel('C:/Users/Romai/OneDrive/Documents/UQAM/BD & ML/Projet micro/dff.xlsx')

# définition des valeurs manquantes
df1.loc[df1['GEN_01'] >= 7, 'GEN_01'] = None
df1.loc[df1['DHHGMS'] > 4, 'DHHGMS'] = None
df1.loc[df1['DHHGHSZ'] > 5, 'DHHGHSZ'] = None
df1.loc[df1['DHH_OWN'] > 2, 'DHH_OWN'] = None
df1.loc[df1['SDCFIMM'] > 2, 'SDCFIMM'] = None
df1.loc[df1['FVCGTOT'] > 3, 'FVCGTOT'] = None
df1.loc[df1['EDUDR04'] > 4, 'EDUDR04'] = None
df1.loc[df1['HWTGBMI'] >= 999.96, 'HWTGBMI'] = None
df1.loc[df1['FLU_160'] > 2, 'FLU_160'] = None
df1.loc[df1['INCG2'] > 4, 'INCG2'] = None
df1.loc[df1['GEN_07'] > 5, 'GEN_07'] = None
df1.loc[df1['HCU_1AA'] > 2, 'HCU_1AA'] = None
df1.loc[df1['PACDPAI'] > 3, 'PACDPAI'] = None
df1.loc[df1['ALCDTTM'] > 3, 'ALCDTTM'] = None
df1.loc[df1['SMK_01A'] > 2, 'SMK_01A'] = None
```

```
df1.boxplot('HWTGBMI')
# pourcentage de valeurs manquantes
df_isna= df1.isna().mean().round(4) * 100

# on supprime les variables avec plus de 10% de valeurs manquantes
df1.drop(['FVCGTOT', 'INCG2'], axis=1, inplace=True)



### Traitement des valeurs manquantes
#On supprime les observations avec au moins une valeur manquante
df2= df1.dropna(thresh=15)




### Traitement valeurs abberantes
import seaborn as sns
sns.catplot(x='GEN_01', y='HWTGBMI', data=df1)
sns.catplot(x='GEN_01', y='HWTGBMI', kind="box", data=df2)

index1 = df2[ (df2.HWTGBMI >= 30) & (df2.GEN_01 ==1) ].index
index2 = df2[ (df2.HWTGBMI <= 18.5) & (df2.GEN_01 ==1) ].index
index3 = df2[ (df2.HWTGBMI >= 30) & (df2.GEN_01 ==2) ].index
index4 = df2[ (df2.HWTGBMI <= 18.5) & (df2.GEN_01 ==2) ].index
index5 = df2[ (df2.HWTGBMI >= 35) & (df2.GEN_01 ==3) ].index
index6 = df2[ (df2.HWTGBMI <= 18.5) & (df2.GEN_01 ==3) ].index
index7 = df2[ (df2.HWTGBMI >= 40) & (df2.GEN_01 ==4) ].index
index8 = df2[ (df2.HWTGBMI <= 17) & (df2.GEN_01 ==4) ].index

df2.drop(index1, inplace=True)
```

```
df2.drop(index2, inplace=True)

df2.drop(index3, inplace=True)

df2.drop(index4, inplace=True)

df2.drop(index5, inplace=True)

df2.drop(index6, inplace=True)

df2.drop(index7, inplace=True)

df2.drop(index8, inplace=True)



# statistiques descriptives

df2['DHH_SEX'].value_counts(normalize=True)

df2['DHHGAGE'].value_counts(normalize=True)

df2['PACDPAI'].value_counts(normalize=True)

df2['ALCDTTM'].value_counts(normalize=True)



type_counts = df2['GEN_01'].value_counts()

dfGEN_01 = pd.DataFrame({'GEN_01': type_counts},

                 index = [1, 2, 3, 4, 5]

                )

dfGEN_01.plot.pie(y='GEN_01', figsize=(10,10), autopct='%1.1f%%')


sns.catplot(x="GEN_01", kind="count",hue="DHH_SEX", data=df2);


pearsoncorr = df2.corr(method='pearson')


# création des variables indicatrices

df2=pd.get_dummies(df2,
columns=['DHHGAGE','DHHGMS','DHHGHSZ','EDUDR04','PACDPAI','ALCDTTM',

                        'DHH_SEX','DHH_OWN','SDCFIMM','FLU_160','HCU_1AA','SMK_01A','GEN_07'])
```

```
# on supprime une catégories pour les variables catégoriques
df2.drop(['DHH_SEX_2','DHH_OWN_2.0','SDCFIMM_2.0','FLU_160_2.0','HCU_1AA_2.0','SMK_01A_2.0',

'DHHGAGE_12','DHHGMS_1.0','DHHGHSZ_1.0','EDUDR04_1.0','PACDPAI_1.0','ALCDTTM_1.0','GEN_07_1.0'],
    axis=1, inplace=True)



#### Elaboration et évaluation des modèles
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet, LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split, KFold, cross_validate, cross_val_score
from sklearn.model_selection import GridSearchCV



# Modèles
X = df2.drop(['GEN_01'], axis=1)
y = df2['GEN_01']



##### Cross validation ######
```

```
kfolds = 4 # 75% train, 25% test
split = KFold(n_splits=kfolds, shuffle=True, random_state=42)



# Logistic regression
Log_reg = LogisticRegression(multi_class='auto')
acc_log_reg = cross_val_score(Log_reg, X, y, scoring="accuracy", cv=split)
mean_acc_log_reg = np.mean(acc_log_reg)
print(mean_acc_log_reg)



# Decision tree
decision_tree= DecisionTreeClassifier()
acc_decision_tree = cross_val_score(decision_tree, X, y, cv=split,scoring="accuracy")
mean_acc_decision_tree = np.mean(acc_decision_tree)



# Random Forest
RFC= RandomForestClassifier()
acc_RFC = cross_val_score(RFC, X, y, cv=split,scoring="accuracy")
mean_acc_RFC = np.mean(acc_decision_tree)

# XGBC
XGBC= XGBClassifier()
acc_XGBC = cross_val_score(XGBC, X, y, cv=split,scoring="accuracy")
mean_acc_XGBC = np.mean(acc_XGBC)

# Gradient boosting
GBC = GradientBoostingClassifier()
```

```
acc_GBC = cross_val_score(GBC, X, y, cv=split,scoring="accuracy")
mean_acc_GBC = np.mean(acc_GBC)



##### Train/Test split ######

# train/test split logit multinomial.
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, shuffle=True, random_state=42)

from sklearn.linear_model import LogisticRegression
#create an instance and fit the model
result = Log_reg.fit(X_train, y_train)



#predictions
Predictions = GBC.predict(X_test)
from sklearn.metrics import classification_report
print(classification_report(y_test,Predictions))

from sklearn import metrics
print(metrics.confusion_matrix(y_test, Predictions))



from sklearn.metrics import accuracy_score

# Logistic regression
Log_reg.fit(X_train,y_train)
```

```
y_pred = Log_reg.predict(X_test)
acc1 = accuracy_score(y_test,y_pred)

# Decision Tree
decision_tree.fit(X_train,y_train)
y_pred2 = decision_tree.predict(X_test)
acc2 = accuracy_score(y_test,y_pred2)

# Random Forest
RFC.fit(X_train,y_train)
y_pred3 = RFC.predict(X_test)
acc3 = accuracy_score(y_test,y_pred3)

# XGBC
XGBC.fit(X_train,y_train)
y_pred4 = XGBC.predict(X_test)
acc4 = accuracy_score(y_test,y_pred4)


# Gradient boosting
GBC.fit(X_train,y_train)
y_pred5 = GBC.predict(X_test)
acc5 = accuracy_score(y_test,y_pred5)



# Alternative 1
y1=df2['GEN_01'].copy()
y1.loc[y1 == 1] = 3
y1.loc[y1 == 2] = 3
```

```
#Grid Search


p_test3 = {'n_estimators':[100,250,500,750,1000,1250,1500,1750],
        'learning_rate':[0.15,0.1,0.05,0.01,0.005,0.001]}

tuning = GridSearchCV(estimator =GradientBoostingClassifier(),
         param_grid = p_test3, scoring='accuracy', cv=split)
tuning.fit(X_train,y_train)


y_pred_acc = tuning.predict(X_test)
print('Accuracy Score : ' + str(accuracy_score(y_test,y_pred_acc)))
```

Ressources

http://fr.ap-hm.fr/site/cso-paca-ouest/obesite/definition

https://www.codespeedy.com/multiclass-classification-using-scikit-learn/

https://lovelyanalytics.com/2017/10/16/grid-search/

https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6

emerj.com/ai-sector-overviews/machine-learning-in-pharma-medicine/

https://www.datacareer.de/blog/parameter-tuning-in-gradient-boosting-gbm/