OBESITY DATASET

Patrick
Théophane
Andréas

ESILV - Python for data analysis - project 2023

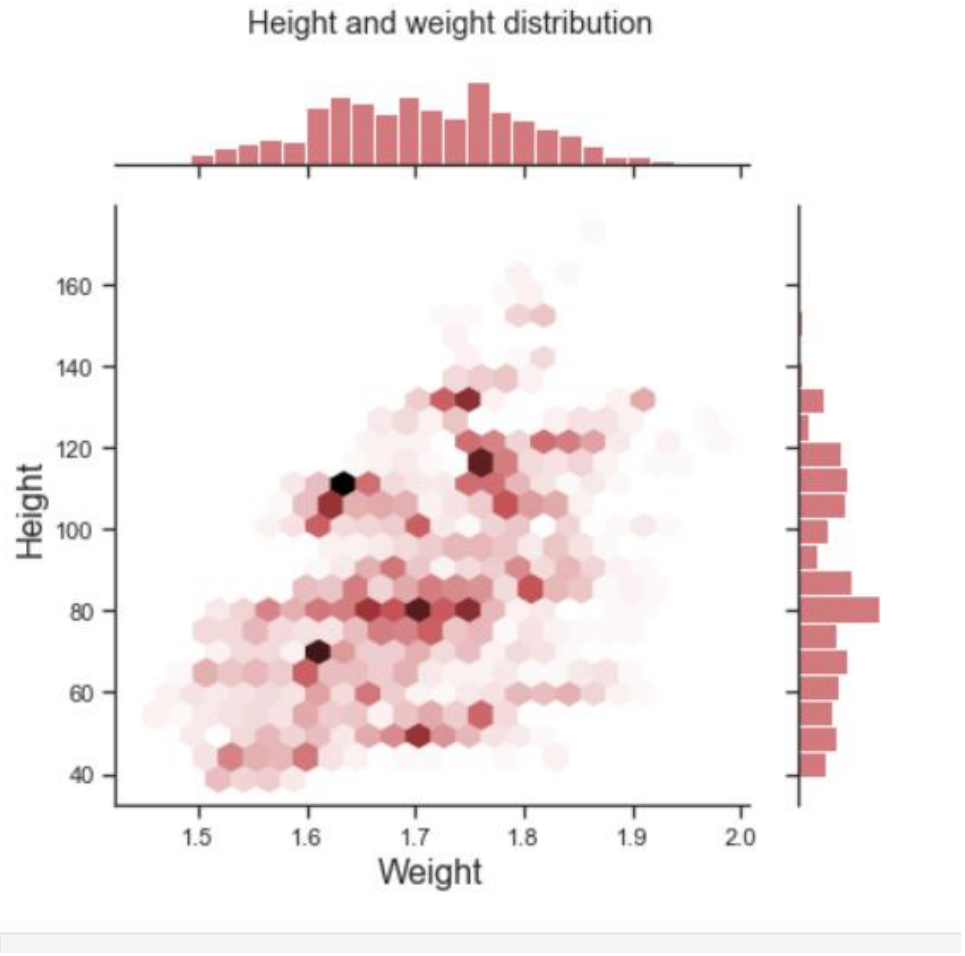
The Dataset

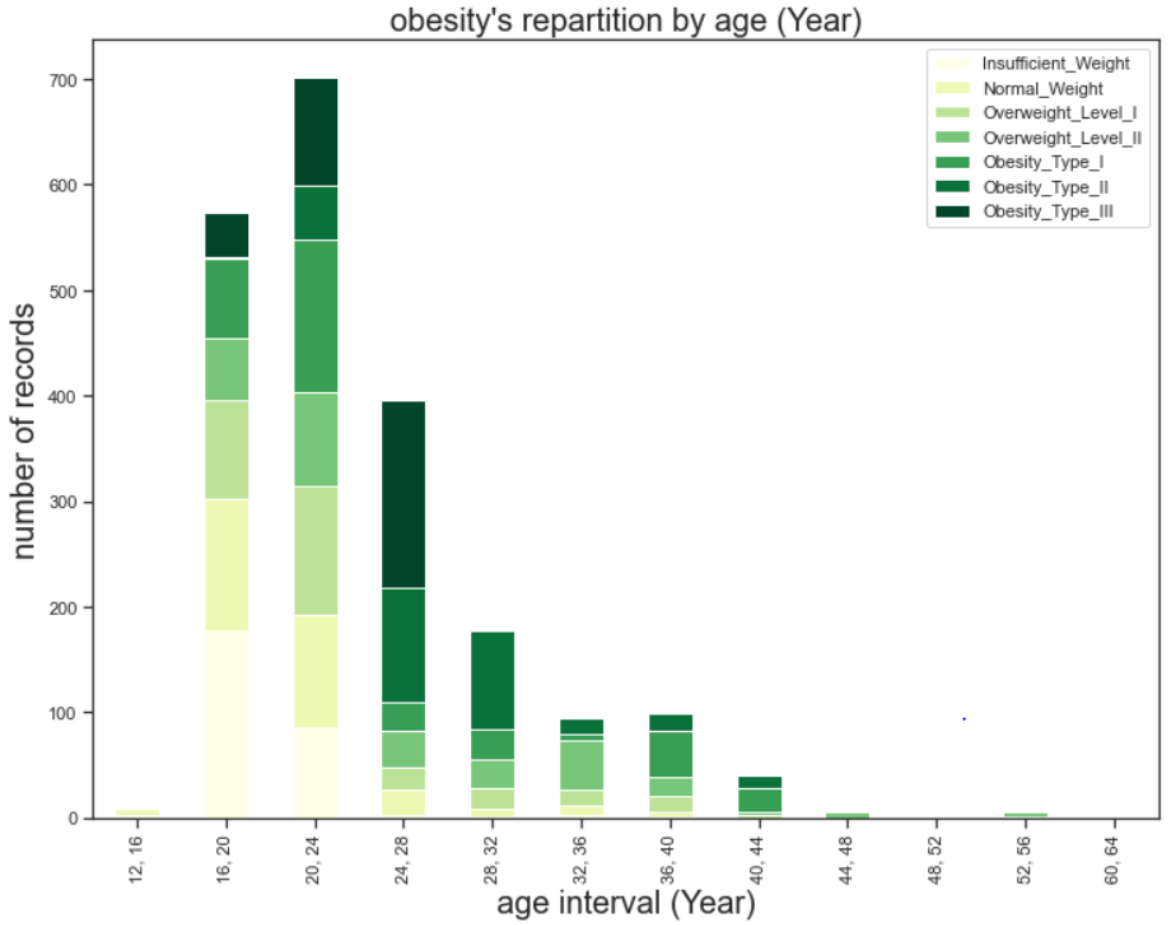
```
RangeIndex: 2111 entries, 0 to 2110
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Gender                2111 non-null   object
 1   Age                   2111 non-null   float64
 2   Height                2111 non-null   float64
 3   Weight                2111 non-null   float64
 4   genetic               2111 non-null   object
 5   high_calorific        2111 non-null   object
 6   vegetable             2111 non-null   float64
 7   frequence_meal        2111 non-null   float64
 8   between_meal          2111 non-null   object
 9   smoke                 2111 non-null   object
 10  water                 2111 non-null   float64
 11  calories_monitoring   2111 non-null   object
 12  physical_activity     2111 non-null   float64
 13  technology_time       2111 non-null   float64
 14  alcohol               2111 non-null   object
 15  transportation        2111 non-null   object
 16  obesity               2111 non-null   object
dtypes: float64(8), object(9)
memory usage: 280.5+ KB
```

OUR DATAS

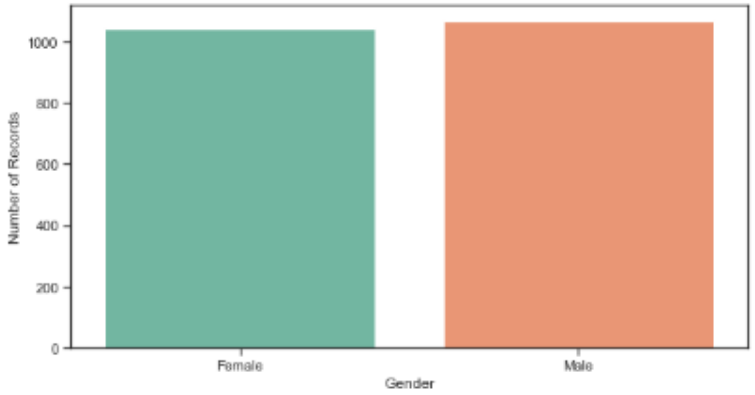
DataViz and Biais

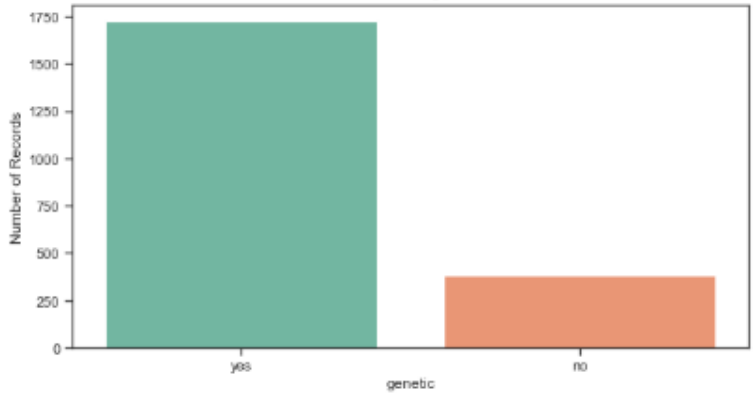
Correlation and biais

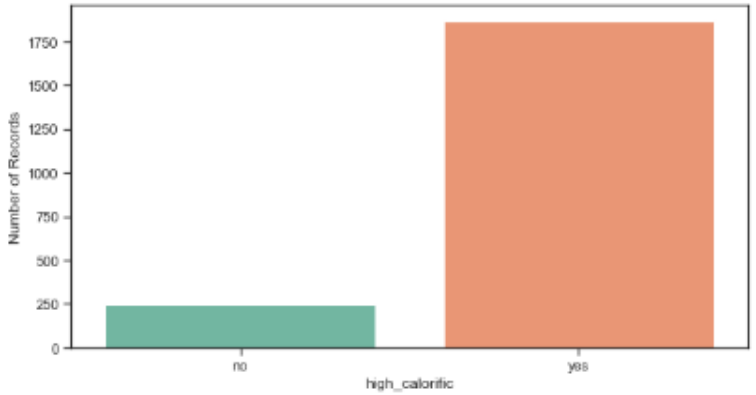


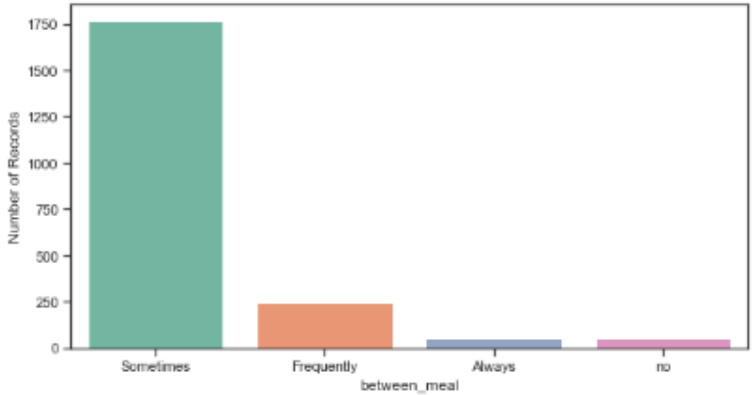


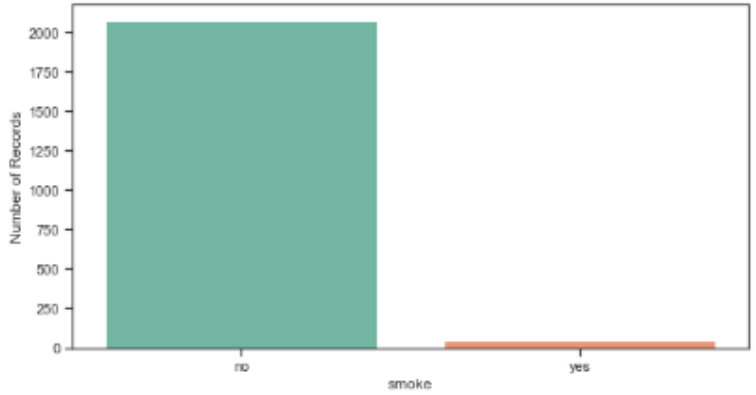
Variables' repartition

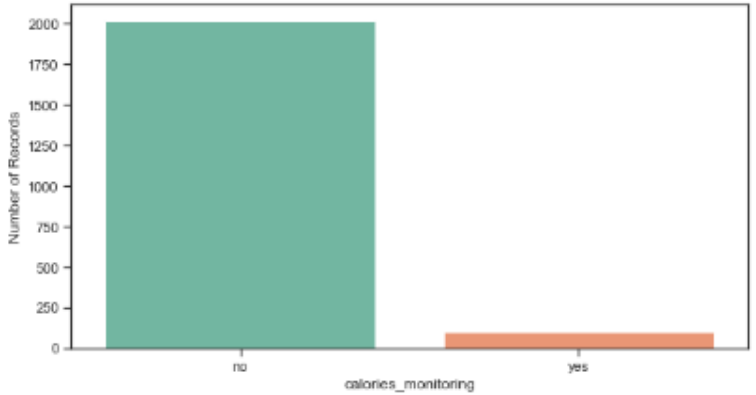


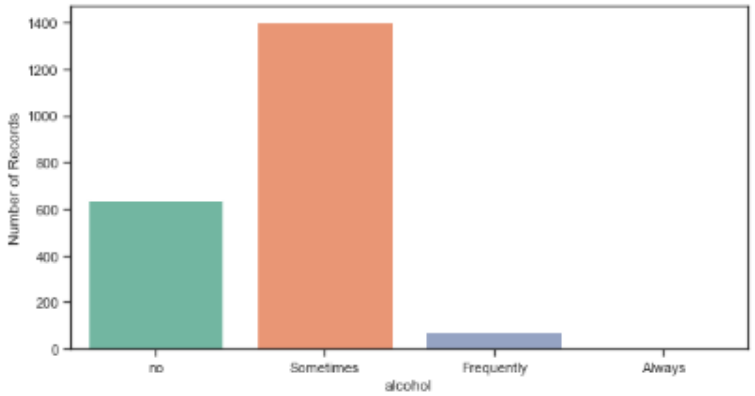


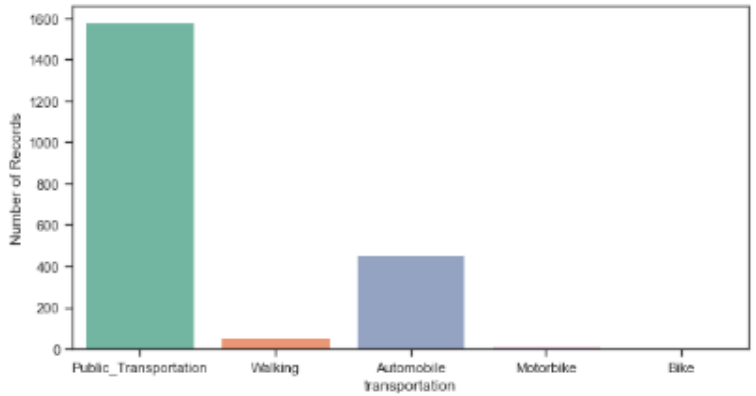


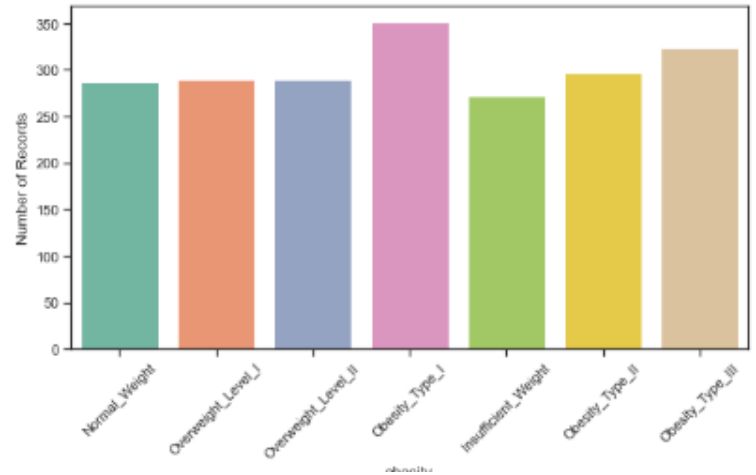




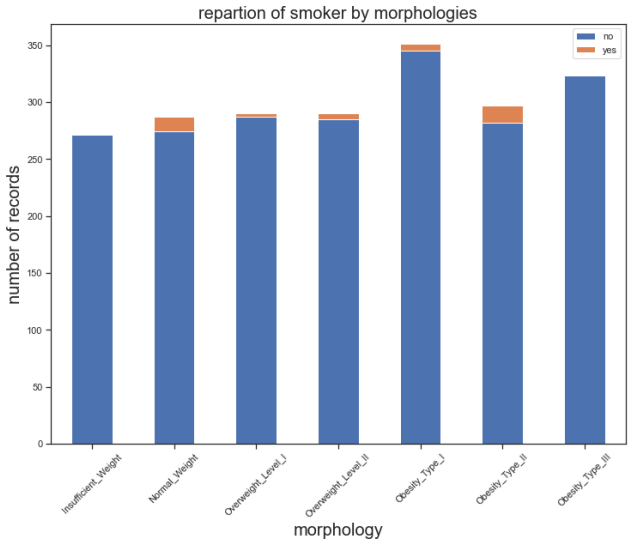


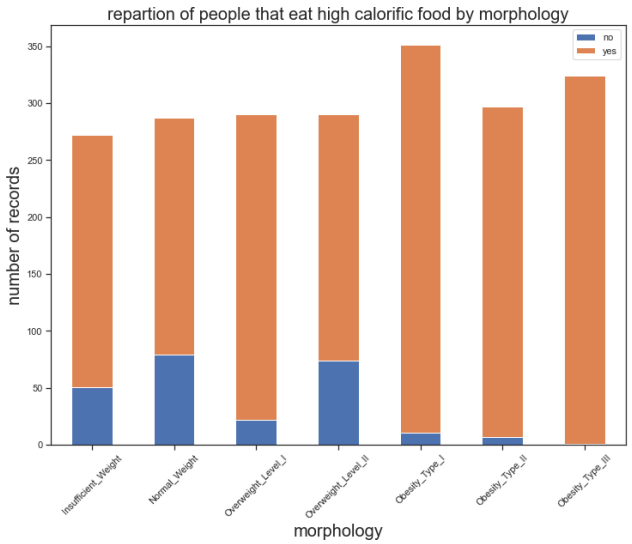


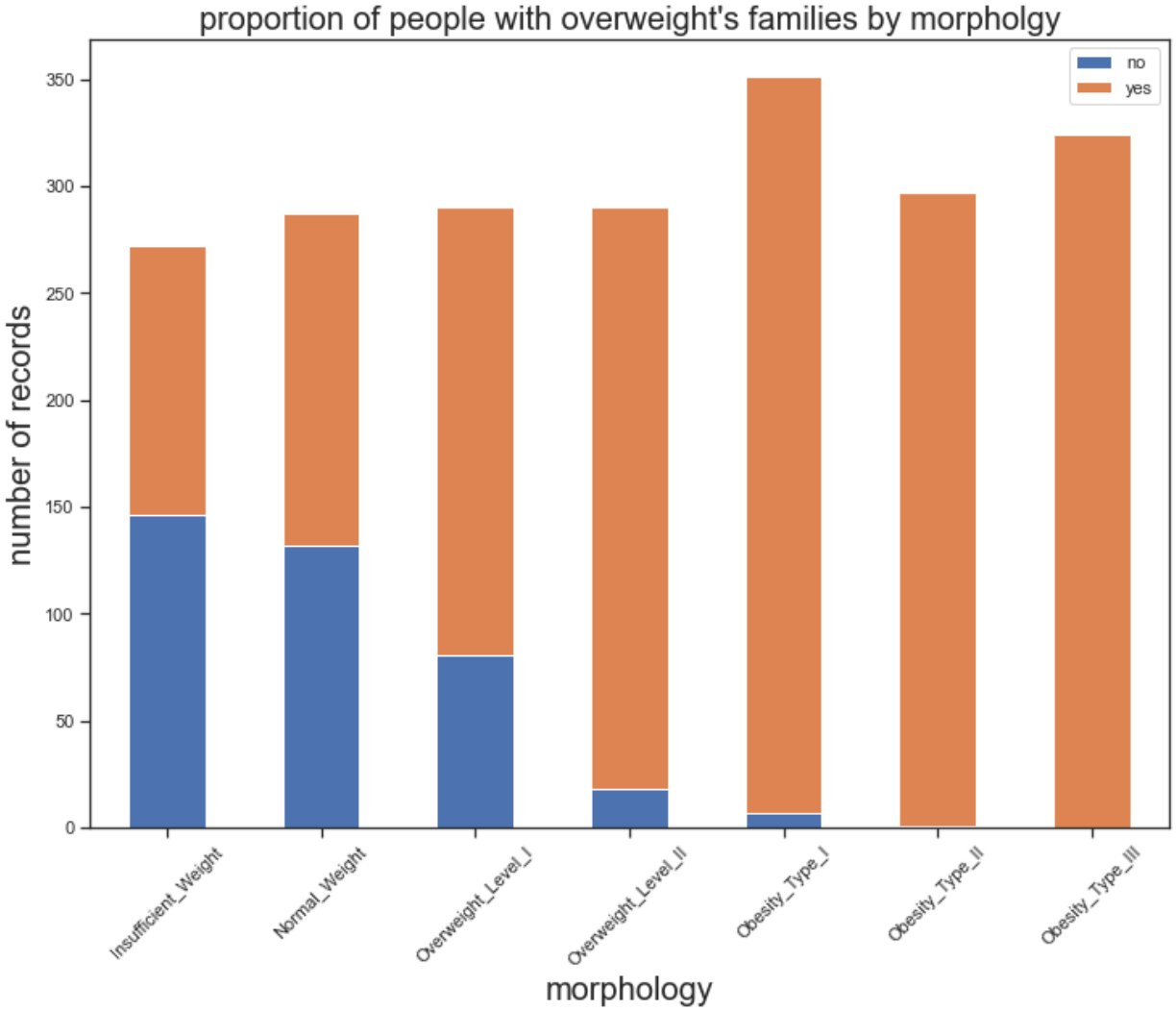


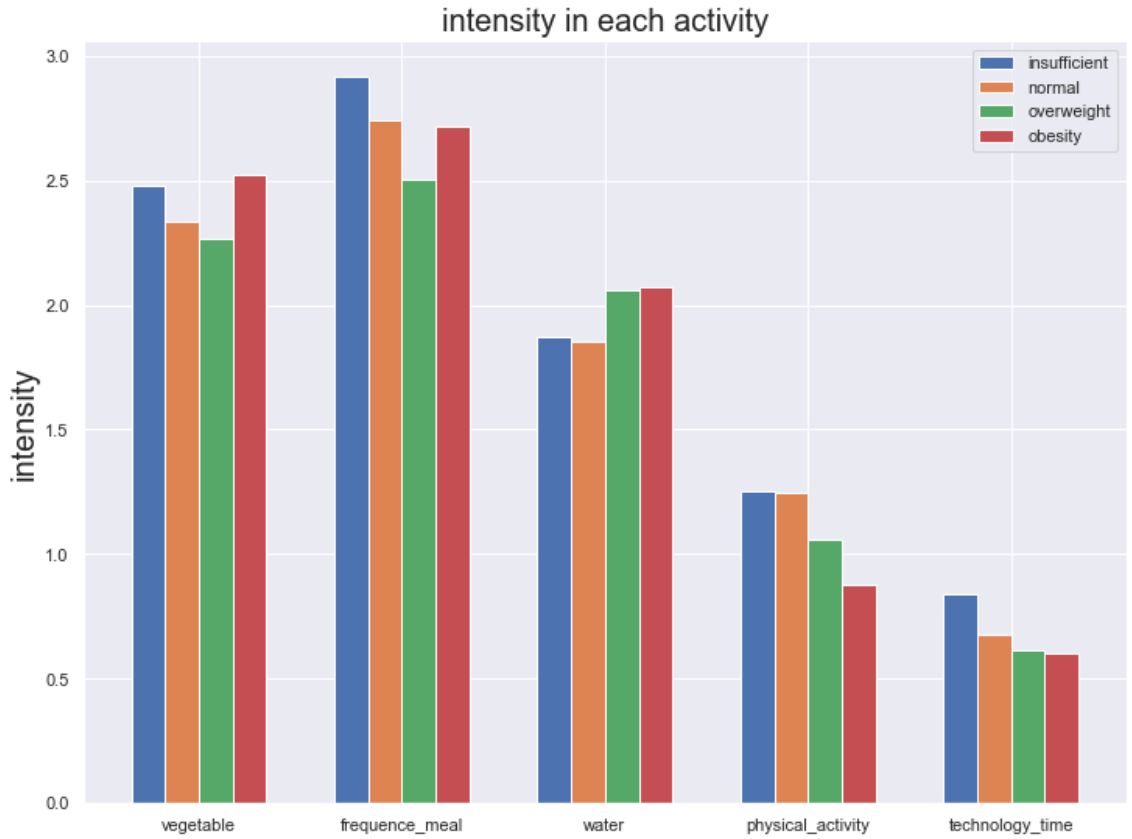


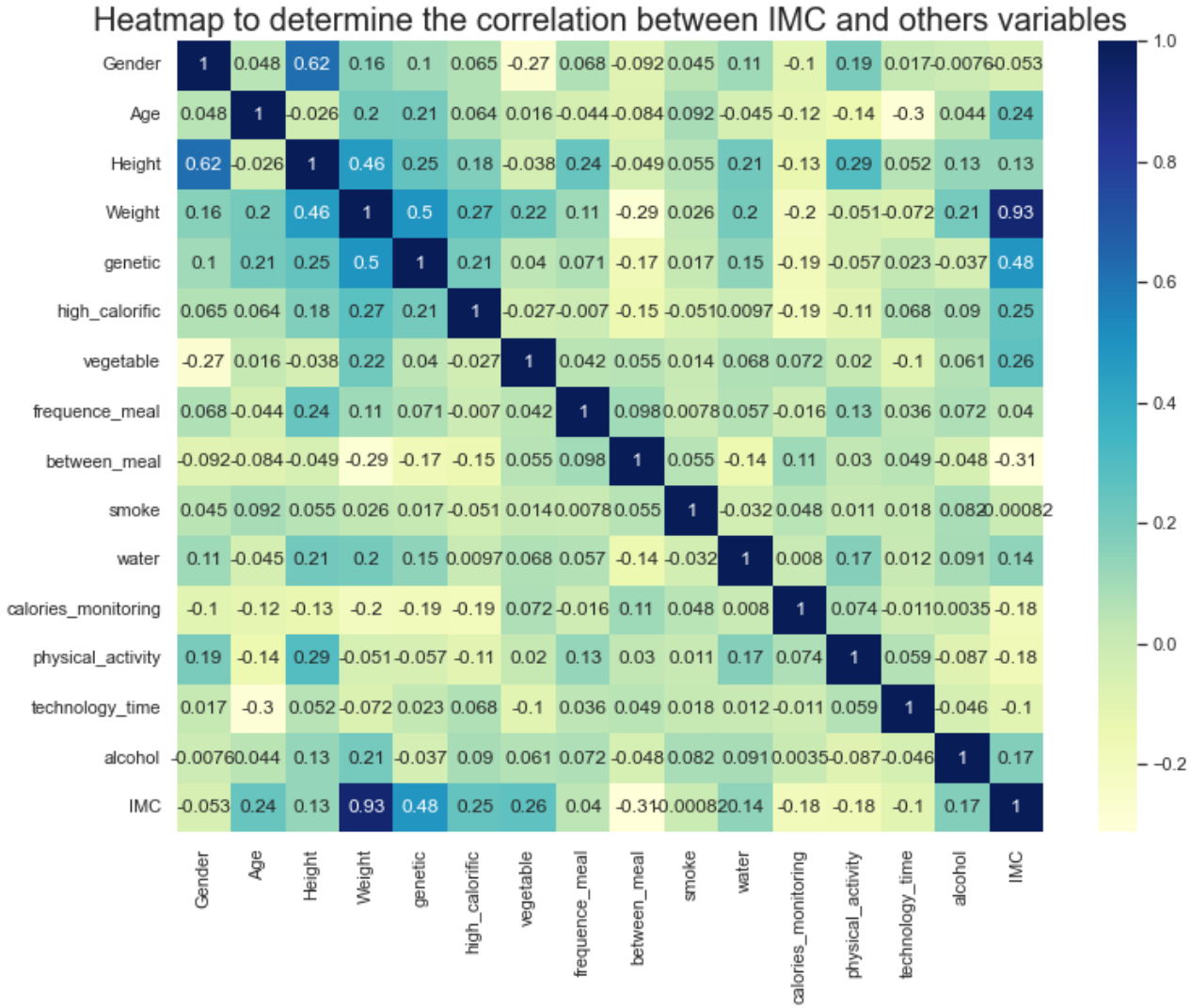
Repartition per Obesity type








intensity in each activity
insufficient
normal
overweight
obesity
intensity
3.0
2.5
2.0
1.5
1.0
0.5
0.0
vegetable
frequence_meal
water
physical_activity
technology_time


Heatmap to determine the correlation between IMC and others variables
Gender 1 0.048 0.62 0.16 0.1 0.065 -0.27 0.068 -0.092 0.045 0.11 -0.1 0.19 0.017 -0.0076 0.053
Age 0.048 1 -0.026 0.2 0.21 0.064 0.016 -0.044 -0.084 0.092 -0.045 -0.12 -0.14 -0.3 0.044 0.24
Height 0.62 -0.026 1 0.46 0.25 0.18 -0.038 0.24 -0.049 0.055 0.21 -0.13 0.29 0.052 0.13 0.13
Weight 0.16 0.2 0.46 1 0.5 0.27 0.22 0.11 -0.29 0.026 0.2 -0.2 -0.051 -0.072 0.21 0.93
genetic 0.1 0.21 0.25 0.5 1 0.21 0.04 0.071 -0.17 0.017 0.15 -0.19 -0.057 0.023 -0.037 0.48
high_calorific 0.065 0.064 0.18 0.27 0.21 1 -0.027 -0.007 -0.15 -0.051 0.0097 -0.19 -0.11 0.068 0.09 0.25
vegetable -0.27 0.016 -0.038 0.22 0.04 -0.027 1 0.042 0.055 0.014 0.068 0.072 0.02 -0.1 0.061 0.26
frequence_meal 0.068 -0.044 0.24 0.11 0.071 -0.007 0.042 1 0.098 0.0078 0.057 -0.016 0.13 0.036 0.072 0.04
between_meal -0.092 -0.084 -0.049 -0.29 -0.17 -0.15 0.055 0.098 1 0.055 -0.14 0.11 0.03 0.049 -0.048 -0.31
smoke 0.045 0.092 0.055 0.026 0.017 -0.051 0.014 0.0078 0.055 1 -0.032 0.048 0.011 0.018 0.082 -0.00082
water 0.11 -0.045 0.21 0.2 0.15 0.0097 0.068 0.057 -0.14 -0.032 1 0.008 0.17 0.012 0.091 0.14
calories_monitoring -0.1 -0.12 -0.13 -0.2 -0.19 -0.19 0.072 -0.016 0.11 0.048 0.008 1 0.074 -0.011 0.0035 -0.18
physical_activity 0.19 -0.14 0.29 -0.051 -0.057 -0.11 0.02 0.13 0.03 0.011 0.17 0.074 1 0.059 -0.087 -0.18
technology_time 0.017 -0.3 0.052 -0.072 0.023 0.068 -0.1 0.036 0.049 0.018 0.012 -0.011 0.059 1 -0.046 -0.1
alcohol -0.0076 0.044 0.13 0.21 -0.037 0.09 0.061 0.072 -0.048 0.082 0.091 0.0035 -0.087 -0.046 1 0.17
IMC -0.053 0.24 0.13 0.93 0.48 0.25 0.26 0.04 -0.31 -0.00082 0.14 -0.18 -0.18 -0.1 0.17 1
Gender
Age
Height
Weight
genetic
high_calorific
vegetable
frequence_meal
between_meal
smoke
water
calories_monitoring
physical_activity
technology_time
alcohol
IMC
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

PREDICTING OBESITY TYPE ?

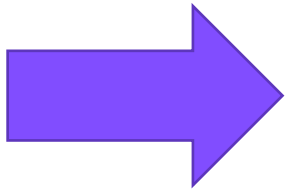
Machine Learning and Preprocessing

Preprocessing

```
8]:

transformer = make_column_transformer(
        (OneHotEncoder(), ["between_meal","alcohol","transportation"]),
        (OrdinalEncoder(), ["Gender","genetic","high_calorific","smoke","calories_monitoring"]),
        remainder=StandardScaler())
transformed = transformer.fit_transform(df)
```

calories_monitoring	physical_activity	technology_time	alcohol	transportation
no	0.0	1.0	no	Public_Transportation
yes	3.0	0.0	Sometimes	Public_Transportation
no	2.0	1.0	Frequently	Public_Transportation
no	2.0	0.0	Frequently	Walking
no	0.0	0.0	Sometimes	Public_Transportation



alcohol_Sometimes	alcohol_no	transportation_Automobile	transportation_Bike	...	smoke	calories_monitoring	Age
0.0	1.0	0.0	0.0	...	0.0	0.0	-0.522124
1.0	0.0	0.0	0.0	...	1.0	1.0	-0.522124
0.0	0.0	0.0	0.0	...	0.0	0.0	-0.206889
0.0	0.0	0.0	0.0	...	0.0	0.0	0.423582
1.0	0.0	0.0	0.0	...	0.0	0.0	-0.364507

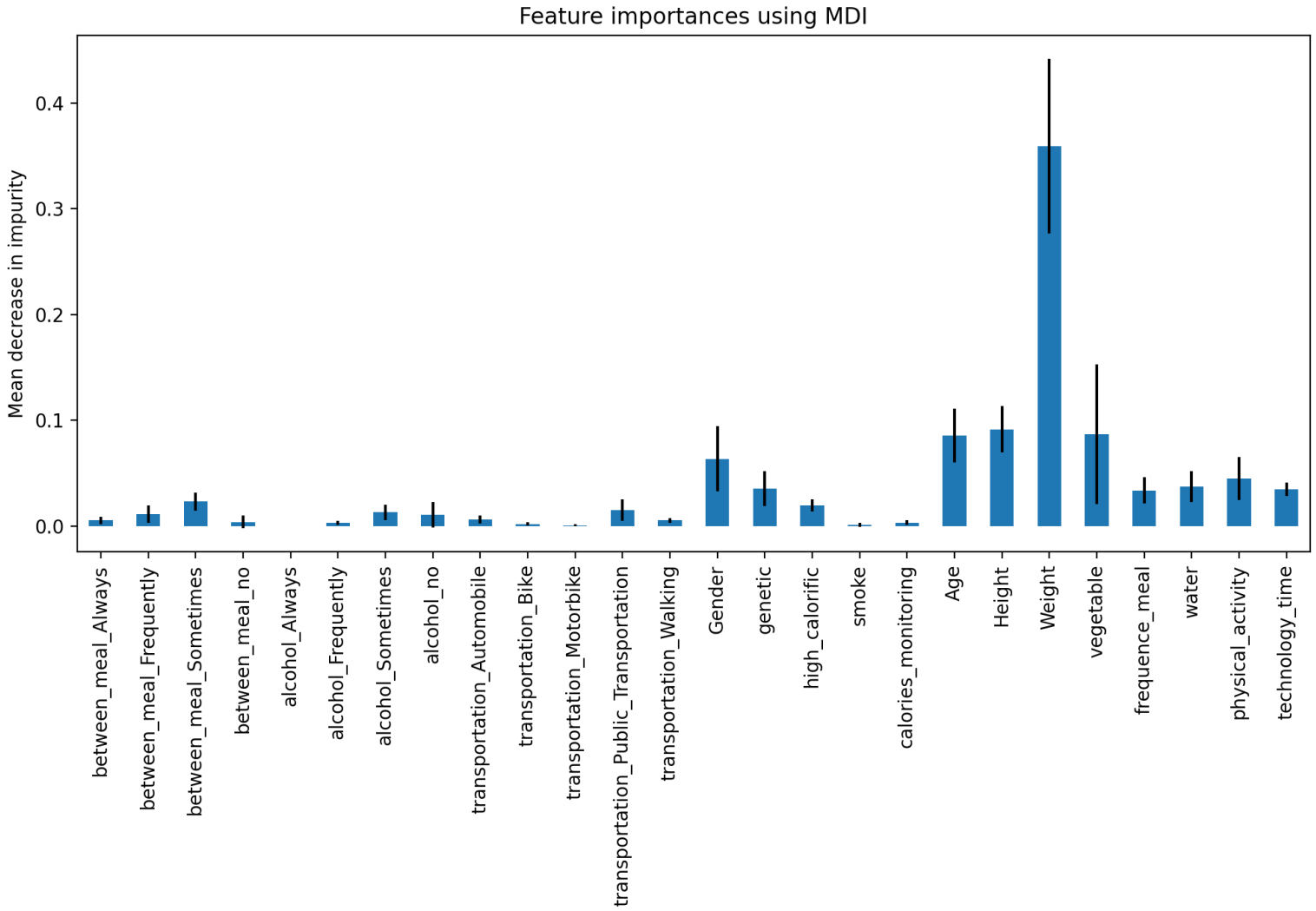
Comparing Machine Learning Performance

```
Test_algo_Obesity(models_liste, labels_liste,X_train, X_test, y_train, y_test)
```

```
Test SVM score = 0.909
Test Desicion tree score = 0.935
Test RandomForest score = 0.938
Test KNeighbors score = 0.833
Test LogisticRegression score = 0.885

best Model for Obesity dataset is RandomForest with test score : 0.938
'RandomForest'
```

Comparing Machine Learning Performance



Improving Machine Learning Performance

```
Test SVM score = 0.924
Test Desicion tree score = 0.924
Test RandomForest score = 0.959
Test KNeighbors score = 0.817
Test LogisticRegression score = 0.872
```

```
best Model for Obesity dataset is RandomForest with test score : 0.959
: 'RandomForest'
```

We see that by removing these 2 columns, we even get a better accuracy

Improving Machine Learning Performance

In [24]:

```
param_grid={"max_depth":[k for k in range(5)]+[None],"n_estimators":[k for k in range(500,501)],"criterion":["gini","entropy","log_loss"]}
grid=GridSearchCV(RandomForestClassifier(),param_grid)
grid.fit(X_train,y_train)
print('grid params best model',grid.best_params_)
print('grid score best model',grid.best_score_)
best_model=grid.best_estimator_
print('test score best model', best_model.score(X_test,y_test))
```

```
grid params best model {'criterion': 'entropy', 'max_depth': None, 'n_estimators': 500}
grid score best model 0.9438181401740724
test score best model 0.9589905362776026
```

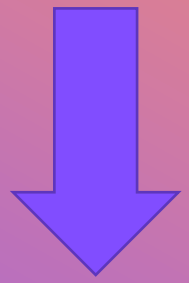
SCALING AND DEPLOYING

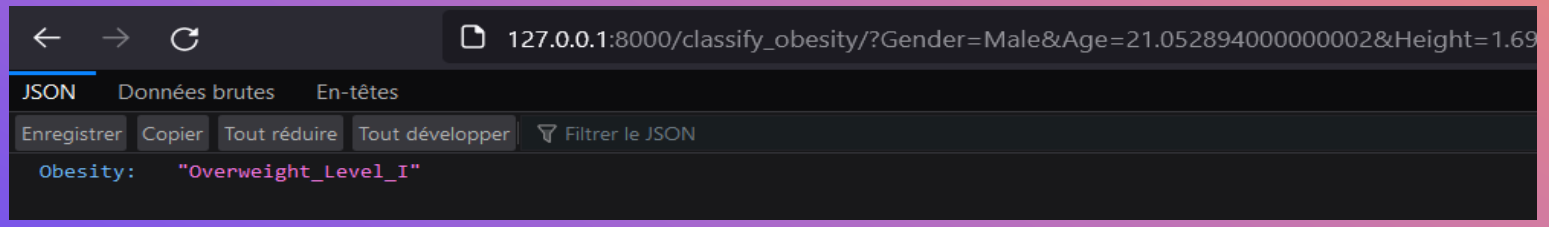
API & Django & Virtual Environnemnt

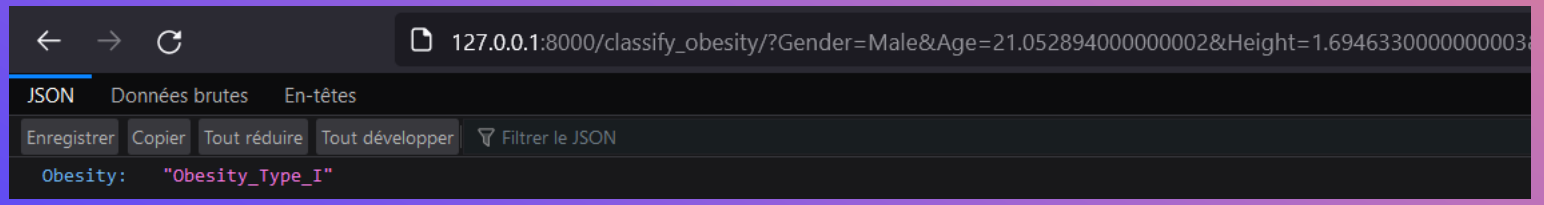
API & Venv


```
(env)
asgiref==3.5.2
backports.zoneinfo==0.2.1
Django==4.1.3
djangorestframework==3.14.0
joblib==1.2.0
numpy==1.23.5
pandas==1.5.2
python-dateutil==2.8.2
pytz==2022.6
scikit-learn==1.1.2
scipy==1.9.3
six==1.16.0
sqlparse==0.4.3
threadpoolctl==3.1.0
tzdata==2022.7
```

```
DjangoMLAPI>python api\manage.py runserver
```







MERCI