Application des Machines à Vecteurs de Support en cardiologie : prévision intelligente des Cardiopathies

Annexe TIPE 2021/2022

Othmane EL HAMDAOUI, SCEI 34098

Juillet 2022

1 Code

```

# In[1]:


import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import hvplot.pandas
from scipy import stats

get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style("whitegrid")
plt.style.use("fivethirtyeight")


# In[2]:
# Importation de la base de donn es



data = pd.read_csv("C:/Users/othma/desktop/heart.csv")
data.head()



# In[3]:
# Visualisation de donn es paer deux param tres pour montrer que
    celle-ci est lin airement non s parable


plt.figure(figsize=(9, 7))
```

```
33
34
35 plt.scatter(data.age[data.target==1],
36             data.thalach[data.target==1],
37             c="red")
38
39
40 plt.scatter(data.age[data.target==0],
41             data.thalach[data.target==0],
42             c="green")
43
44
45 plt.title("")
46 plt.xlabel("Age")
47 plt.ylabel("Fr quence cardiaque maximale atteinte")
48 plt.legend(["Disease", "No Disease"]);
49
50
51 # In[4]:
52
53
54 from sklearn.metrics import accuracy_score, confusion_matrix,
       classification_report
55
56 def print_score(clf, X_train, y_train, X_test, y_test, train=True):
57     if train:
58         pred = clf.predict(X_train)
59         clf_report = pd.DataFrame(classification_report(y_train,
       pred, output_dict=True))
60         print("Train Result:\n
       ================================================")
61         print(f"Accuracy Score: {accuracy_score(y_train, pred) *
       100:.2f}%")
62         print("_______________________________________________")
63         print(f"CLASSIFICATION REPORT:\n{clf_report}")
64         print("_______________________________________________")
65         print(f"Confusion Matrix: \n {confusion_matrix(y_train,
       pred)}\n")
66
67     elif train==False:
68         pred = clf.predict(X_test)
69         clf_report = pd.DataFrame(classification_report(y_test,
       pred, output_dict=True))
70         print("Test Result:\n
       ================================================")
71         print(f"Accuracy Score: {accuracy_score(y_test, pred) *
       100:.2f}%")
72         print("_______________________________________________")
73         print(f"CLASSIFICATION REPORT:\n{clf_report}")
74         print("_______________________________________________")
75         print(f"Confusion Matrix: \n {confusion_matrix(y_test, pred
       )}\n")
76
77
78 # In[5]:
79 # Division de la base de donn es en deux sous bases, une pour
       l  entrainement   et   l autre   pour le test de   l  entrainement
```

```

from sklearn.model_selection import train_test_split

X = dataset.drop('target', axis=1)
y = dataset.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
    =0.3, random_state=42)


# In[6]:
# Construction du mod le de la classification

from sklearn.svm import SVC


svm_clf = SVC(kernel='rbf', gamma=0.1, C=1.0)
svm_clf.fit(X_train, y_train)

print_score(svm_clf, X_train, y_train, X_test, y_test, train=True)
print_score(svm_clf, X_train, y_train, X_test, y_test, train=False)


# Out[6]:

Train Result:
================================================
Accuracy Score: 95.40%
------------------------------------------------
CLASSIFICATION REPORT:
                     0           1  accuracy   macro avg  weighted
    avg
precision     0.969419    0.941026  0.953975    0.955222
    0.954490
recall        0.932353    0.973475  0.953975    0.952914
    0.953975
f1-score      0.950525    0.956975  0.953975    0.953750
    0.953916
support     340.000000  377.000000  0.953975  717.000000
    717.000000
------------------------------------------------
Confusion Matrix:
 [[317  23]
 [ 10 367]]

Test Result:
================================================
Accuracy Score: 90.26%
------------------------------------------------
CLASSIFICATION REPORT:
                     0           1  accuracy   macro avg  weighted
    avg
precision     0.944828    0.865031  0.902597    0.904929
    0.906225
recall        0.861635    0.946309  0.902597    0.903972
    0.902597
```

```
f1-score      0.901316    0.903846   0.902597    0.902581
    0.902540
support     159.000000  149.000000   0.902597  308.000000
    308.000000
_________________________________________________
Confusion Matrix:
 [[137  22]
 [  8 141]]


# In[7]:
# Mod le  apr s  tunning


from sklearn.model_selection import GridSearchCV

svm_clf = SVC(kernel='rbf', gamma=0.1, C=1.0)

params = {"C":(0.1, 0.5, 1, 2, 5, 10, 20),
          "gamma":(0.001, 0.01, 0.1, 0.25, 0.5, 0.75, 1),
          "kernel":('linear', 'poly', 'rbf')}

svm_cv = GridSearchCV(svm_clf, params, n_jobs=-1, cv=5, verbose=1,
    scoring="accuracy")
svm_cv.fit(X_train, y_train)
best_params = svm_cv.best_params_
print(f"Best params: {best_params}")

svm_clf = SVC(**best_params)
svm_clf.fit(X_train, y_train)

print_score(svm_clf, X_train, y_train, X_test, y_test, train=True)
print_score(svm_clf, X_train, y_train, X_test, y_test, train=False)


# Out[7]:

Fitting 5 folds for each of 147 candidates, totalling 735 fits
Best params: {'C': 2, 'gamma': 0.5, 'kernel': 'rbf'}
Train Result:
================================================
Accuracy Score: 100.00%
_________________________________________________
CLASSIFICATION REPORT:
                0      1  accuracy  macro avg  weighted avg
precision     1.0    1.0       1.0        1.0           1.0
recall        1.0    1.0       1.0        1.0           1.0
f1-score      1.0    1.0       1.0        1.0           1.0
support     340.0  377.0       1.0      717.0         717.0
_________________________________________________
Confusion Matrix:
 [[340   0]
 [  0 377]]

Test Result:
================================================
Accuracy Score: 98.05%
```

```
------------------------------------------------
CLASSIFICATION REPORT:
                    0           1  accuracy   macro avg  weighted avg
precision    0.963636    1.000000  0.980519    0.981818     0.981228
recall       1.000000    0.959732  0.980519    0.979866     0.980519
f1-score     0.981481    0.979452  0.980519    0.980467     0.980500
support    159.000000  149.000000  0.980519  308.000000   308.000000
------------------------------------------------
Confusion Matrix:
 [[159   0]
 [  6 143]]
```

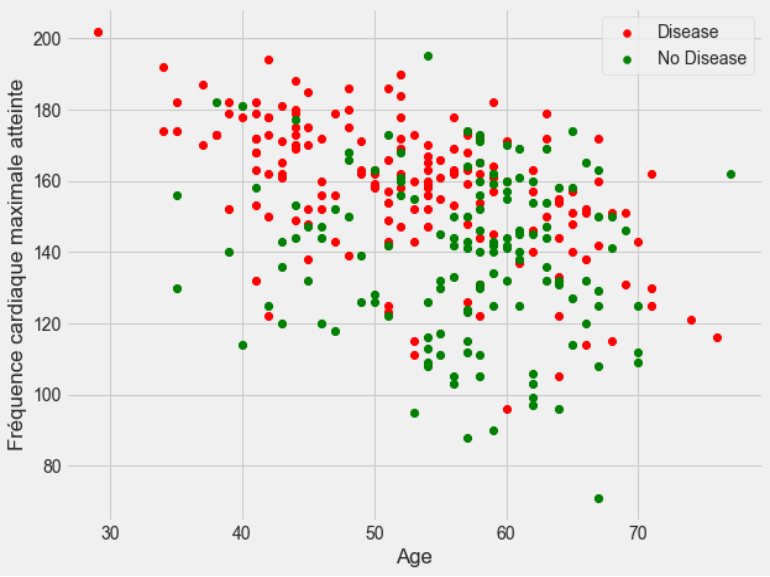

Figure 1: Out[3]: Visualisation de données par deux paramètres pour montrer que celle-ci est linéairement non séparables