**Báo cáo Svm kernel**

## TẦNG BUS

### SVM KERNEL

#### Cơ sở lý thuyết:

Thuật toán SVM KERNEL là việc áp dụng svm lên bài toán mà dữ liệu giữa các class là hoàn toàn không linear separable

#### Bảng hàm và lớp:

Bảng 1: Hàm

| STT | Tên lớp | Mục đích của lớp | File lưu trữ | Tên sinh viên phụ trách |
| --- | --- | --- | --- | --- |
| 1 | Ket Qua  Output:các Parameters tốt nhất | Sử dụng hàm gridsearch để tìm tập parameters tốt cho giải thuật | svmKernel.pyx | Đinh Công Minh |
| 2 | Ve  Output: Vẽ ra tập 2 feature của dataset | Xem sự phân bố các lable trong tập bộ vẽ | svmKernel.pyx | Đinh Công Minh |

#### Mã nguồn

##### **Các thư viện và trỏ liên kết tới DAO**

# coding=utf-8

import sys

sys.path.append('../DAO/')

# tap du lieu su dung

import dataProcessing as dp

# thu vien ve cua python

import matplotlib.pyplot as plt

#thu vien ho tro svm kernel

from sklearn.svm import SVC

# chia tap du lieu ban dau thanh 2 tap la training va testing

from sklearn.model\_selection import train\_test\_split

from sklearn.model\_selection import GridSearchCV

#xu ly matrix

import numpy as np

#Đánh giá

from sklearn.metrics import classification\_report

from sklearn import metrics

**#chon cột dữ liệu**

x\_train\_CoLoc, x\_test\_CoLoc, y\_train\_CoLoc, y\_test\_CoLoc=train\_test\_split(data\_CoLoc(0,['tuoi','nghe\_nghiep','hon\_nhan','hoc\_van','co\_the\_tin\_dung','co\_nha\_o','vay\_ca\_nhan','kenh\_lien\_lac','thang\_lien\_lac','ngay\_lien\_lac','thoi\_luong\_lien\_lac']),data\_CoLoc(1,None),test\_size=0.2)

x\_train\_KhongLoc, x\_test\_KhongLoc, y\_train\_KhongLoc, y\_test\_KhongLoc=train\_test\_split(data\_khongLoc(0,['tuoi','nghe\_nghiep','hon\_nhan','hoc\_van','co\_the\_tin\_dung','co\_nha\_o','vay\_ca\_nhan','kenh\_lien\_lac','thang\_lien\_lac','ngay\_lien\_lac','thoi\_luong\_lien\_lac']),data\_khongLoc(1,None),test\_size=0.2)

##### Lớp ket Qua:

tuned\_parameters = [{'kernel': ['rbf'], 'gamma': [0.001, 0.01, 0.1, 1],

'C': [0.01, 0.1, 1, 10]}]

grid = GridSearchCV(SVC(), tuned\_parameters, cv=5)

grid.fit(x\_train\_CoLoc, y\_train\_CoLoc)

print("parameter tốt nhất có lọc %s with a score of %0.2f"% (grid.best\_params\_, grid.best\_score\_))

clf = SVC(kernel='poly',gamma=0.01,C=10,degree=3) # rbf Kernel

clf.fit(x\_train\_CoLoc, y\_train\_CoLoc)

y\_pred = clf.predict(x\_test\_CoLoc)

print("Accuracy:",metrics.accuracy\_score(y\_test\_CoLoc, y\_pred))

#tìm parameter tốt nhất khong lọc

grid1 = GridSearchCV(SVC(), tuned\_parameters, cv=5)

grid1.fit(x\_train\_KhongLoc, y\_train\_KhongLoc)

print("parameter tốt nhất khong lọc %s with a score of %0.2f"% (grid1.best\_params\_, grid1.best\_score\_))

clf = SVC(kernel='rbf',gamma=0.01,C=10) # rbf Kernel

clf.fit(x\_train\_KhongLoc, y\_train\_KhongLoc)

y\_pred = clf.predict(x\_test\_KhongLoc)

print("Accuracy:",metrics.accuracy\_score(y\_test\_KhongLoc, y\_pred))

##### Lớp Ve:

X=data\_CoLoc(0,['tuoi','nghe\_nghiep']) # Lấy hai thuộc tính đầu tiên

Y=data\_CoLoc(1,None)

X=np.array(X)

Y=np.array(Y)

X\_min, X\_max = X[:, 0].min() - .5, X[: ,0].max() + .5

Y\_min, Y\_max = X[: ,1].min() - .5, X[: ,1].max() + .5

plt.figure(2, figsize=(8, 6))

plt.clf()

#Biểu diễn tập dữ liệu huấn luyện bằng đồ

plt.scatter(X[:, 0], X[:,1], c=Y,cmap=plt.cm.Paired)

plt.xlabel('tuoi')

plt.ylabel('nghe\_nghiep')

plt.xlim(X\_min, X\_max)

plt.ylim(Y\_min, Y\_max)

plt.xticks(())

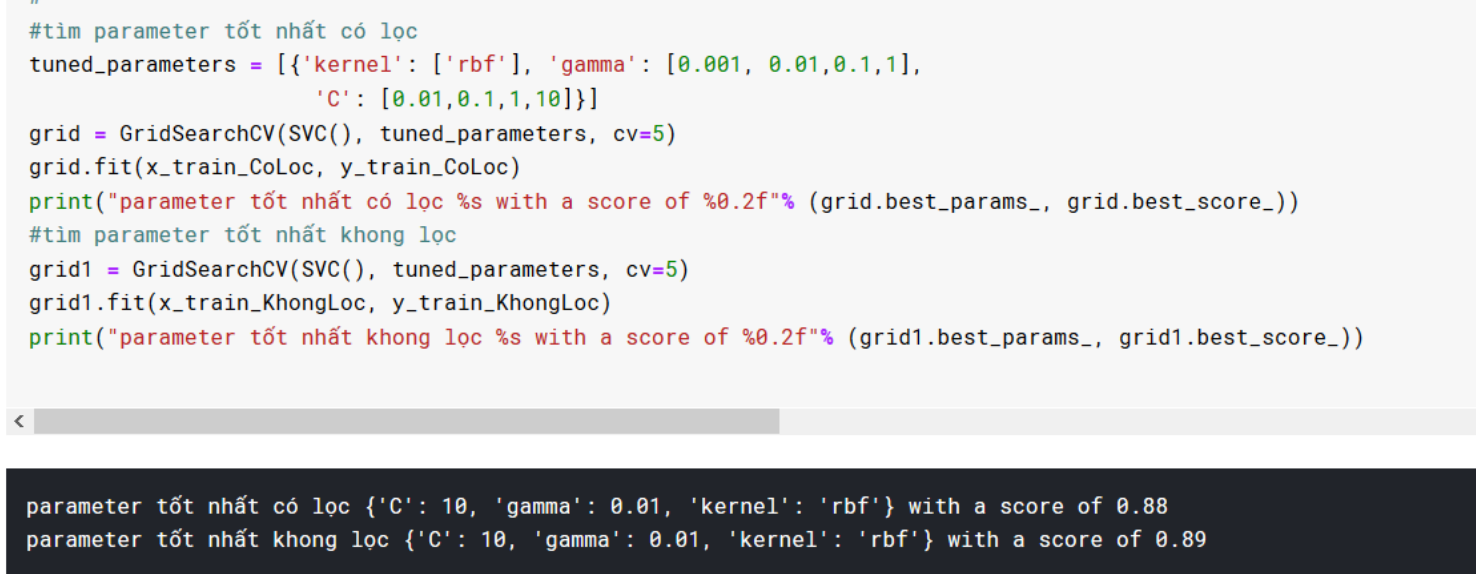
plt.yticks(())

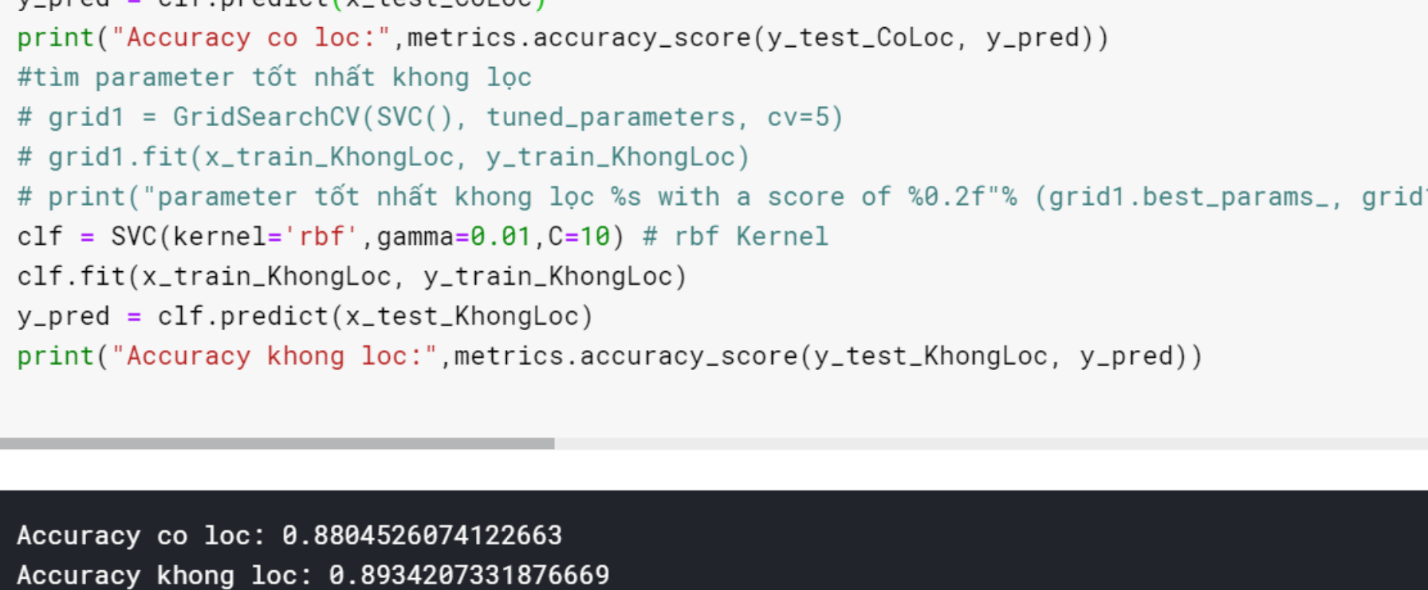
plt.show()

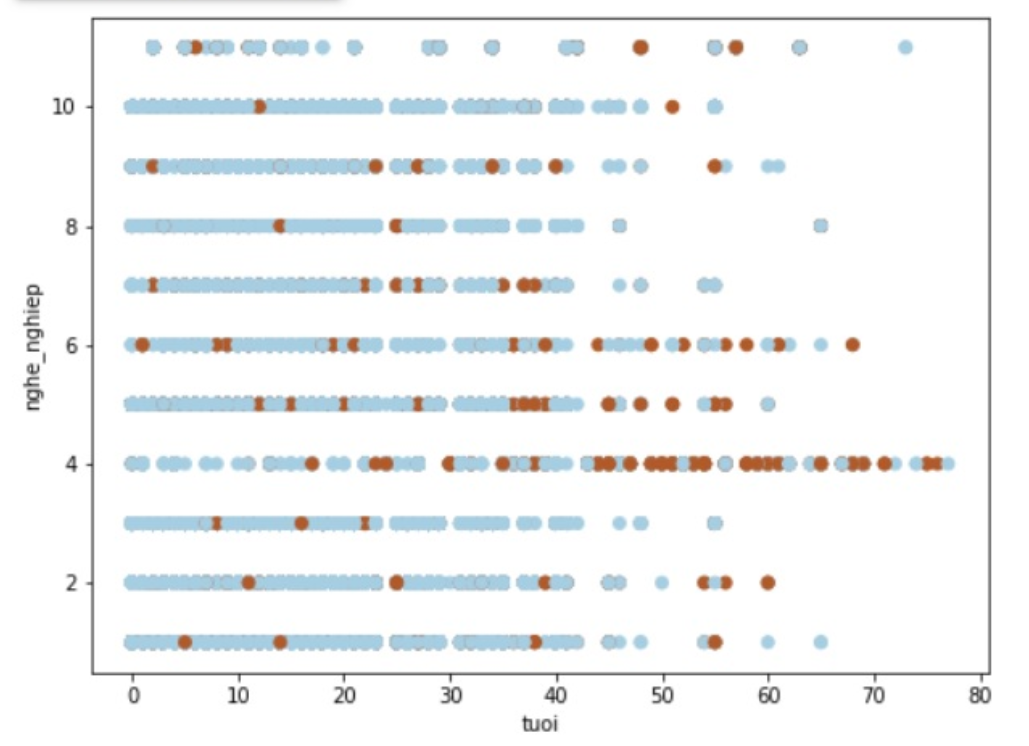
1. **Đánh giá**

**Các cột em chọn gồm:**

*'tuoi','nghe\_nghiep','hon\_nhan','hoc\_van','co\_the\_tin\_dung','co\_nha\_o','vay\_ca\_nhan','kenh\_lien\_lac','thang\_lien\_lac','ngay\_lien\_lac','thoi\_luong\_lien\_lac'.*

****

****

****

**Màu nâu : 1**

**Màu xanh : 0**