**Đề bài:** Sử dụng sklearn huấn luyện SVM tập dữ liệu digits 8\*8

Dùng paint tạo số 0..9 việt tay, mỗi số trong ô vuông 100\*100 nền đen, lưu vào máy.

Đọc ảnh vào xử lý về dạng 8\*8 rồi nhận dạng. Báo cáo độ chính xác

**Code Python:**

import matplotlib.pyplot as plt

import numpy as np

import time

import datetime as dt

import matplotlib.pyplot as plt

import cv2

import glob

from sklearn import datasets, svm, metrics

from sklearn.datasets import fetch\_mldata

from mnist\_helpers import \*

(images, targets) = datasets.load\_digits(n\_class=10, return\_X\_y=True)

show\_some\_digits(images,targets)

X\_data = images/255.0

Y = targets

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

X\_train, \_, y\_train, \_ = train\_test\_split(X\_data, Y, test\_size=0.2, random\_state=42)

################ Classifier with good params ###########

# Create a classifier: a support vector classifier

param\_C = 10

param\_gamma = 0.05

classifier = svm.SVC(kernel = 'linear',C=param\_C,gamma=param\_gamma)

#We learn the digits on train part

start\_time = dt.datetime.now()

print('Start learning at {}'.format(str(start\_time)))

classifier.fit(X\_train, y\_train)

end\_time = dt.datetime.now()

print('Stop learning {}'.format(str(end\_time)))

elapsed\_time= end\_time - start\_time

print('Elapsed learning {}'.format(str(elapsed\_time)))

test\_image = cv2.imread('image/0-80.png')

test\_image = cv2.cvtColor(test\_image, cv2.COLOR\_BGR2GRAY)

test\_image = cv2.resize(test\_image, (8,8))

test\_image = test\_image.reshape(-1,64)

X\_test = test\_image

y\_test = [2]

for name in glob.glob('image/\*.png'):

test\_image = cv2.imread(name)

test\_image = cv2.cvtColor(test\_image, cv2.COLOR\_BGR2GRAY)

test\_image = cv2.resize(test\_image, (8,8))

test\_image = test\_image.reshape(-1,64)

X\_test = np.append(X\_test, test\_image, axis=0)

y\_test = np.append(y\_test, int(name[6]))

expected = y\_test

predicted = classifier.predict(X\_test)

show\_some\_digits(X\_test,predicted,title\_text="Predicted {}")

print("Classification report for classifier %s:\n%s\n"

% (classifier, metrics.classification\_report(expected, predicted)))

cm = metrics.confusion\_matrix(expected, predicted)

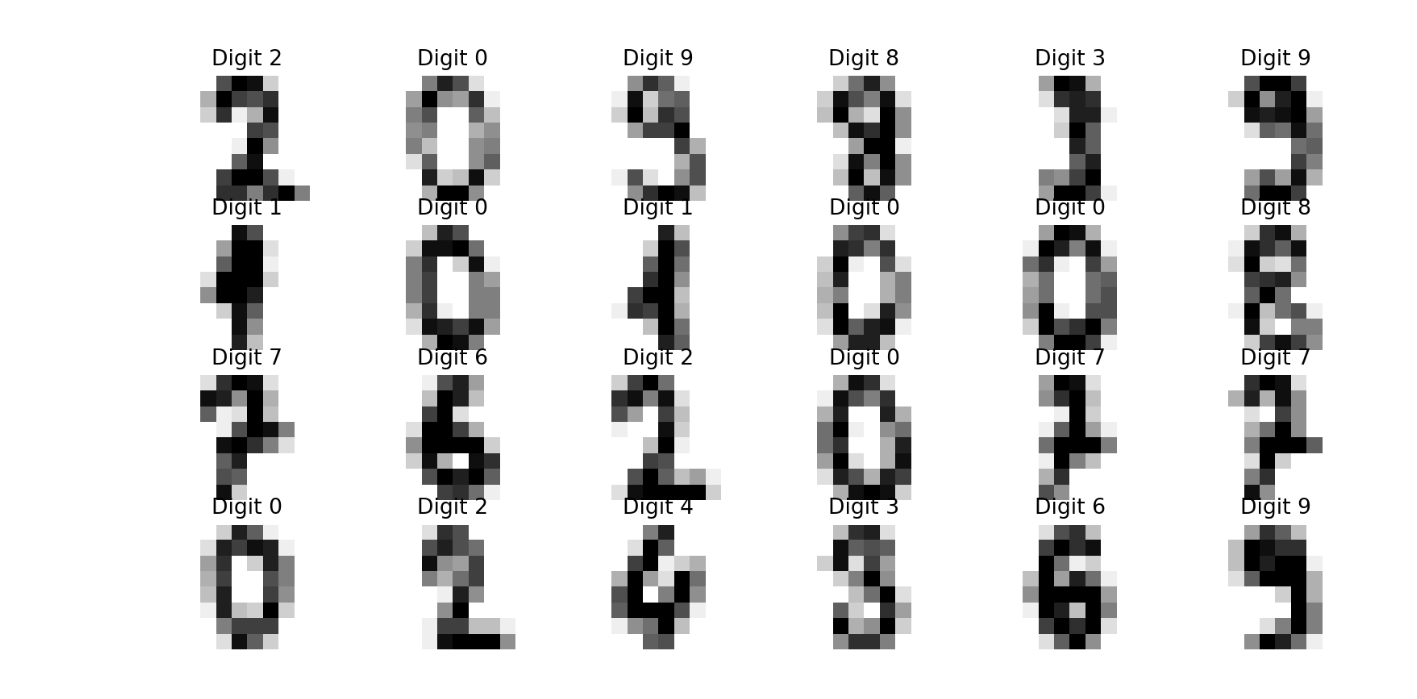
print("Confusion matrix:\n%s" % cm)

plot\_confusion\_matrix(cm)

print("Accuracy={}".format(metrics.accuracy\_score(expected, predicted)))

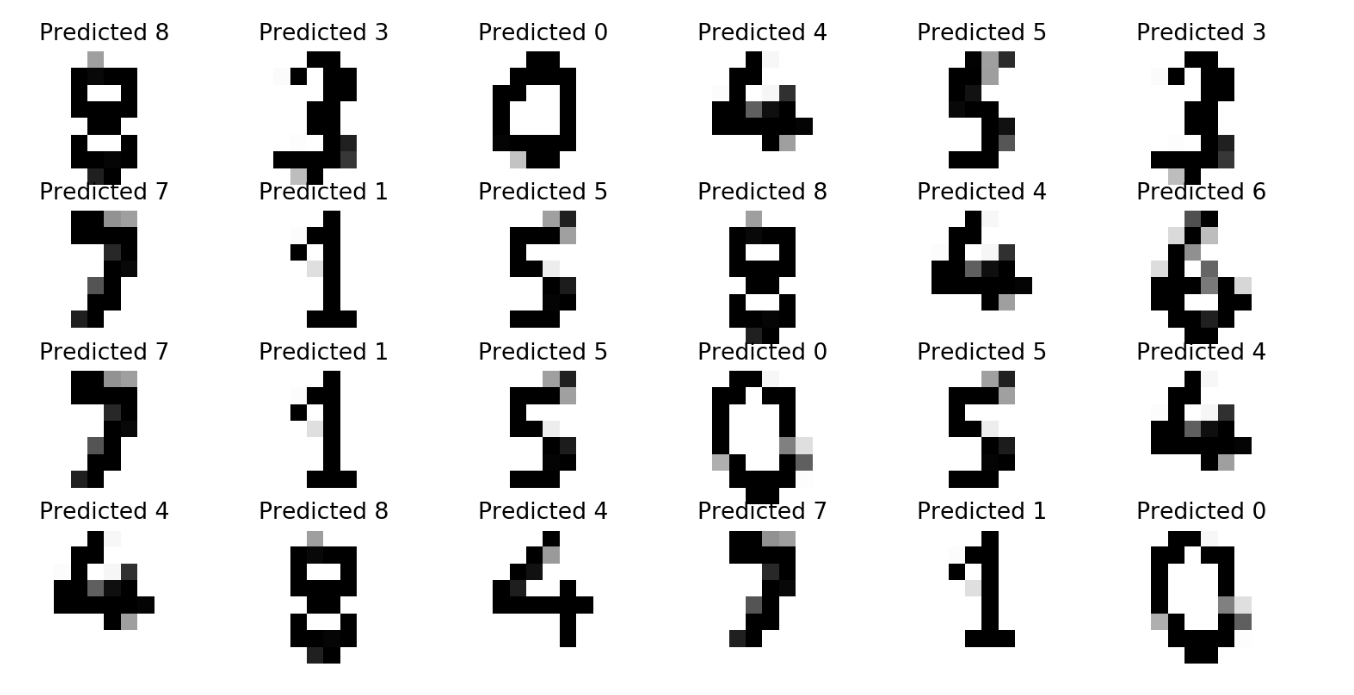
**Kết quả:**

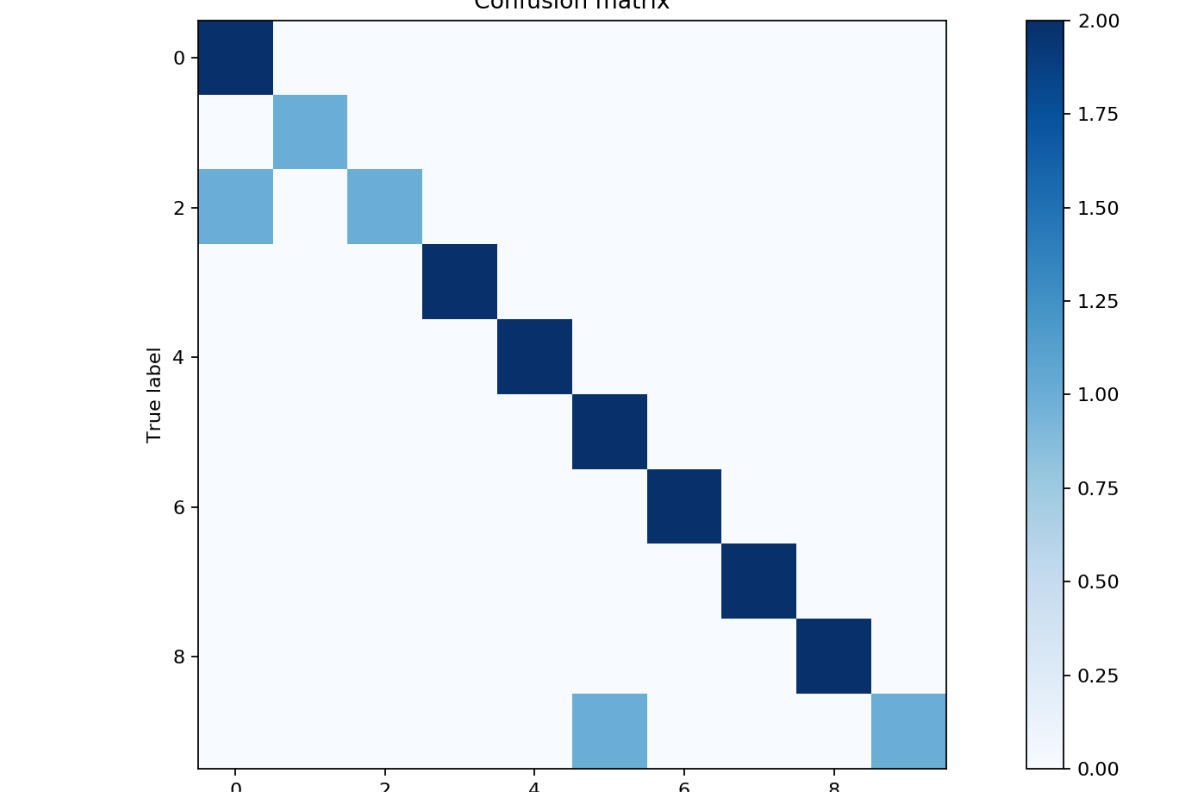
* Show một số mẫu trong tập train:

**-**

-Tập test tạo từ paint 

- Kết quả trên tập test:



* Biểu đồ đánh giá độ chính xác:
* 
* Độ chính xác của tập test (accuracy) = 0.8947368421052632