ep4

November 21, 2018

```
In [2]: import numpy as np
        import matplotlib.pyplot as plt
```

1 MAC0460/5832 - Lista 4: SVM - MNIST

1.0.1 Data de Entrega: 23h55m do dia XX/12/2018

Classificação de dígitos Os dataset para esta tarefa é uma adaptação do disponível na competição do kaggle de reconhecimento de dígitos (https://www.kaggle.com/c/digit-recognizer) e está disponível em http://vision.ime.usp.br/~caiomr/mac0460_5832/train_svm.csv.gz. O dataset está sob a licença Creative Commons Attribution-Share Alike 3.0 license (https://creativecommons.org/licenses/by-sa/3.0/). O dataset foi zipado, e apenas os dígitos 5 e 6 foram mantidos. Cada linha (amostra) do arquivo contém 257 colunas: a primeira informa o label da amostra (0 para o dígito 5, 1 para o dígito 6) e as outras 256 são os valores dos pixels da imagem (16 x 16) que representa o dígito.

Note que esse dataset difere do usado no EP3: as imagens sofreram pequenas rotações e translações aleatórias, além de terem sido escalonadas para o tamanho 16x16. Veja também que pode ser necessário realizar algum tipo de normalização para realizar um treinamento efetivo com SVM. Para auxiliar na normalização dos dados, consultem o seguinte link: http://scikit-learn.org/stable/modules/preprocessing.html.

Q1. Use SVM para classificar os dígitos 5 e 6. Utilize as funções do scikit learn (http://scikit-learn.org/, http://scikit-learn.org/stable/modules/svm.html) para realizar o treinamento.

Teste os kernels linear e RBF da seguinte maneira: 1. Escolha aleatoriamente 932 amostras para formarem o conjunto de teste. 2. Com as 7000 amostras restantes, utilize validação cruzada (com número de folds K = 5) para escolher os parâmetros do seu classificador, isto é: C (peso da *soft margin*) para o kernel linear; C e gamma para o kernel RBF. 3. Plote a curva experimental de aprendizado para o melhor SVM com kernel linear e o melhor SVM com kernel RBF escolhidos por validação cruzada. Use as 932 amostras do conjunto de teste para estimar E_{out}. Comente sobre o resultado.

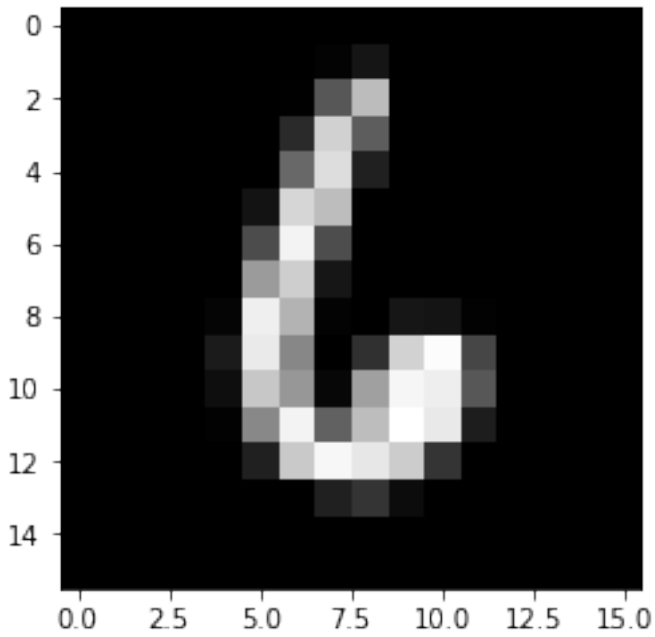
```
In [2]: data = np.genfromtxt('data/train_svm.csv', delimiter=',')
        print(data.shape)
```

```
(7932, 257)
```

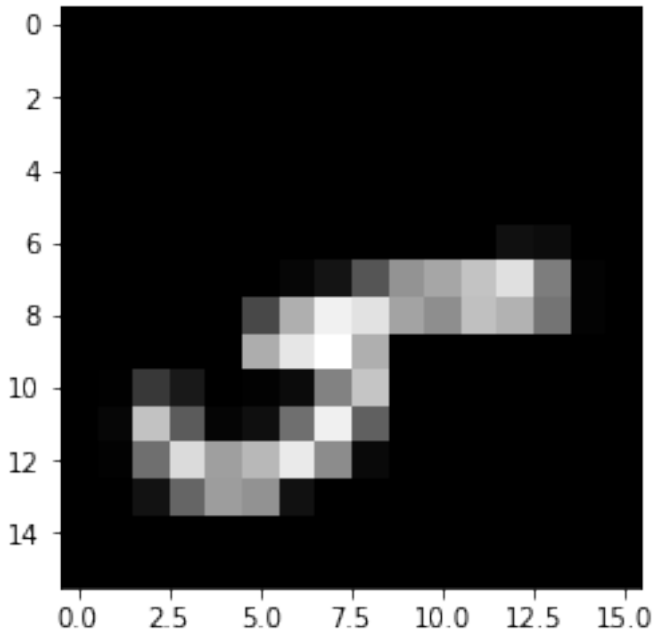
```
In [4]: sample = data[0]
        print("Label: ", int(sample[0]))
```

```
plt.imshow(sample[1:].reshape((16, 16)), cmap='gray')
plt.show()

sample = data[1]
print("Label: ", int(sample[0]))
plt.imshow(sample[1:].reshape((16, 16)), cmap='gray')
plt.show()

sample = data[20]
print("Label: ", int(sample[0]))
plt.imshow(sample[1:].reshape((16, 16)), cmap='gray')
plt.show()

sample = data[25]
print("Label: ", int(sample[0]))
plt.imshow(sample[1:].reshape((16, 16)), cmap='gray')
plt.show()
```

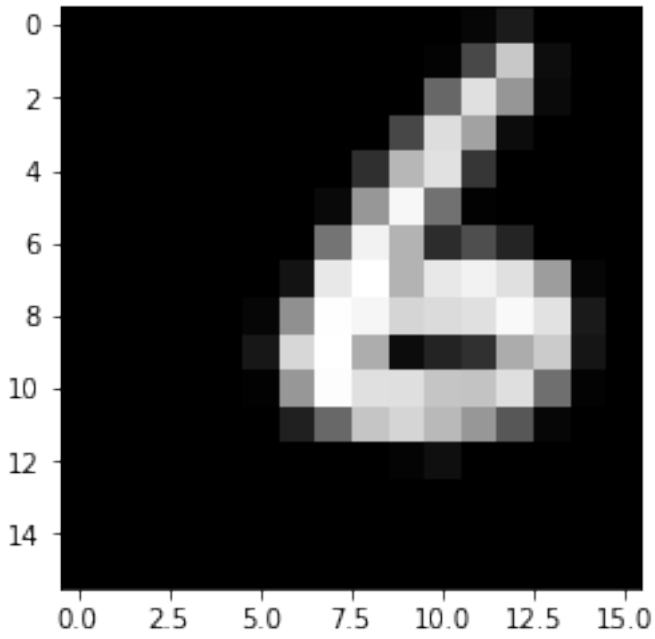
```
Label:  1
```




```
Label:  0
```


0
2
4
6
8
10
12
14
0.0
2.5
5.0
7.5
10.0
12.5
15.0

```
Label:  1
```


0
2
4
6
8
10
12
14
0.0
2.5
5.0
7.5
10.0
12.5
15.0

```
Label:  0
```