SVM 小作业

211870287 丁旭

2023 年 11 月 15 日

题目 1. 使用 python 语言构建一个 SVM 的分类器，对指定数据集进行分类。SVM 算法可采用一些工具库（如 scikit-learn 等）完成，必须具有可视化的分析

解答. 实现过程和结果：

首先使用 numpy 的随机函数生成初始数据，将随机种子设置为 0 确保每次生成的数据一样：

```
np.random.seed(0)
X = np.r_[np.random.randn(20, 2) - [2, 2],
          np.random.randn(20, 2) + [2, 2]]
Y = [-1] * 20 + [1] * 20
```

图 1: 准备分类数据

建立、训练SVM模型：

使用 sklearn 库提供的 SVC 函数将核函数设置为‘linear’类型

```
# 建立、训练SVM模型
clf = svm.SVC(kernel='linear')
clf.fit(X, Y)
```

图 2: 开始训练

获得划分超平面：
划分超平面原方程：w0x0 + w1x1 + b = 0
将其转化为点斜式方程，并把 x0 看作 x，x1 看作 y，b 看作 w2
点斜式：y = -(w0/w1)x - (w2/w1)

```
# 获得划分超平面
# 划分超平面原方程：w0x0 + w1x1 + b = 0
# 将其转化为点斜式方程，并把 x0 看作 x，x1 看作 y，b 看作 w2
# 点斜式：y = -(w0/w1)x - (w2/w1)
w = clf.coef_[0]  # w 是一个二维数据，coef 就是 w = [w0,w1]
a = -w[0] / w[1]  # 斜率
xx = np.linspace(-5, 5)  # 从 -5 到 5 产生一些连续的值（随机的）
# .intercept[0] 获得 bias，即 b 的值，b / w[1] 是截距
yy = a * xx - (clf.intercept_[0]) / w[1]  # 带入 x 的值，获得直线方程
```

画出和划分超平面平行且经过支持向量的两条线（斜率相同，截距不同）

```
b = clf.support_vectors_[0]  # 取出第一个支持向量点
yy_down = a * xx + (b[1] - a * b[0])
b = clf.support_vectors_[-1]  # 取出最后一个支持向量点
yy_up = a * xx + (b[1] - a * b[0])

# 查看相关的参数值
print("w: ", w)
print("a: ", a)
print("support_vectors_: ", clf.support_vectors_)
print("clf.coef_: ", clf.coef_)
```

```
w:  [0.90230696 0.64821811]
a:  -1.391980476255765
support_vectors_:  [[-1.02126202  0.2408932 ]
 [-0.46722079 -0.53064123]
 [ 0.95144703  0.57998206]]
clf.coef_:  [[0.90230696 0.64821811]]
```

绘制划分超平面，边际平面和样本点：

使用matplotlib 进行结果的可视化

```
# 绘制划分超平面，边际平面和样本点
pl.plot(xx, yy, 'k-')
pl.plot(xx, yy_down, 'k--')
pl.plot(xx, yy_up, 'k--')
# 圈出支持向量
pl.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1],
           s=80, facecolors='none')
pl.scatter(X[:, 0], X[:, 1], c=Y, cmap=pl.cm.Paired)

pl.axis('tight')
pl.show()
```

图 3: 结果可视化

分类结果：黑色直线代表超平面，圈出来的点代表支持向量在二维平面中的表示。

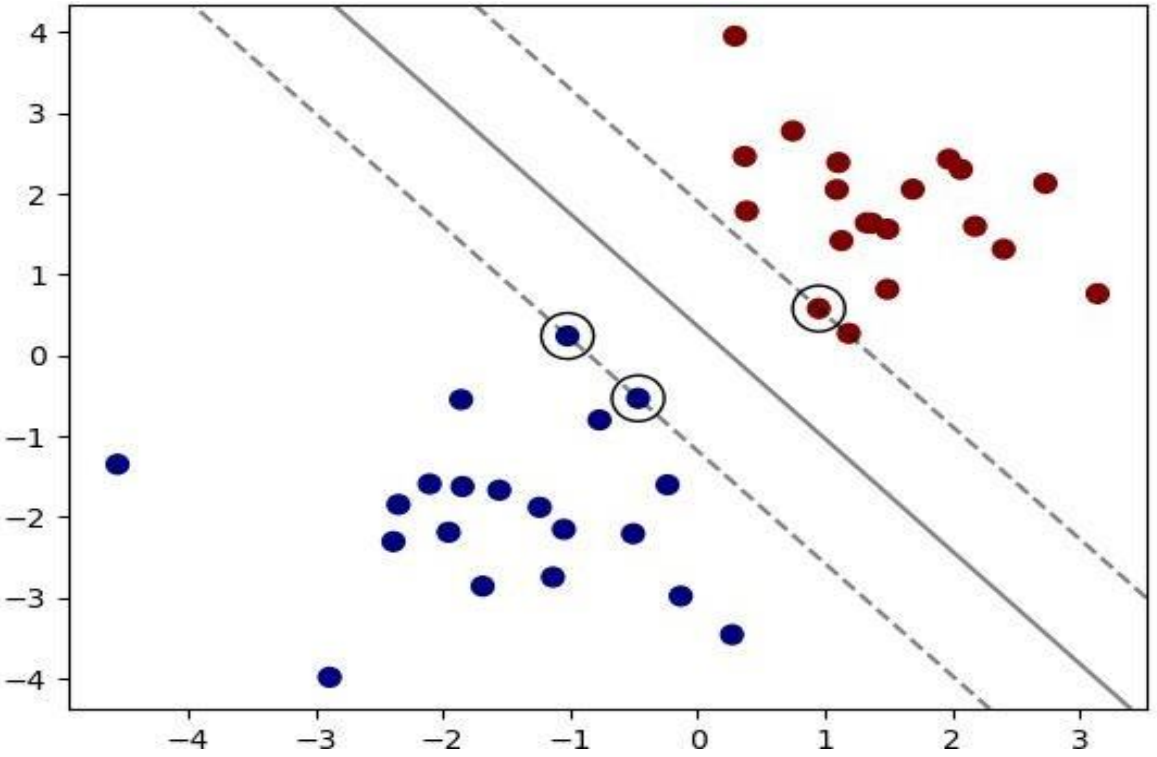

图 4: 分类结果