机器学习 第四次作业

SVM 作业

201250076 袁家乐

2023 年 4 月 21 日

一、SVM 算法原理阐述

对于支持向量机而言，一个点（样例）对应的 margin 是其到分界超平面的垂直距离，具有最小间隔的点称为支持向量，SVM 的核心要义是最大化（所有训练样本的）最小间隔 margin。

本次实验是使用 SVM 实现分类器，重点阐述该原理如下：

对于多特征的分类任务，首先可以将样本从原始空间映射到一个更高维的特征空间，使样本在这个特征空间线性可分。只要原始空间的维度是有限的，则这样的一个高维特征空间必然存在。

通过内积的方式厘清线性和非线性间的联系。

使用核函数进行特征映射，本次实验中使用的是线性核函数，核函数为：

$$K(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{x}^T\boldsymbol{y}$$

对偶形式为：

$$\underset{\boldsymbol{a}}{\operatorname{argmax}} \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j K(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

分类边界为：

$$\boldsymbol{w} = \sum_{i=1}^{n} a_i\, y_i \phi(\boldsymbol{x}_i)$$

通过如下方式进行分类和预测：

$$\boldsymbol{w}^T\phi(\boldsymbol{x}) = \phi(\boldsymbol{x})^T\left(\sum_{i=1}^{n} a_i\, y_i \phi(\boldsymbol{x}_i)\right) = \sum_{i=1}^{n} a_i\, y_i K(\boldsymbol{x},\boldsymbol{x}_i)$$

二、实验步骤阐述

1、数据集获取和预处理

通过 sklearn 库获取 Iris 鸢尾花数据集，分别获取其样本特征和标签，按照 7:3 的比例划分训练集和测试集，将划分情况输出到控制台。

```
# 获取iris数据集
iris = load_iris()

# 获取样本特征和标签
data = iris.data
target = iris.target

# 划分训练集和测试集
data_train, data_test, target_train, target_test = train_test_split(data, target, test_size=0.3, random_state=0)
print("total size:", len(target), "train size:", len(target_train), "test_size:", len(target_test))
```

2、模型训练

通过 sklearn 库 SVM 来构造分类器，设置其核函数为线性，建立并训练模型。

```
# 训练分类器
model = SVC(kernel="linear")
model.fit(data_train, target_train)
```

3、测试和评估

在构造出的模型上运行测试集，计算 micro 和 macro 两种模式下的精确率和召回率，输出到控制台。

```
# 测试和评估
result = model.predict(data_test)
# 精确率
print("micro precision score:", precision_score(target_test, result, average="micro"))
print("macro precision score:", precision_score(target_test, result, average="macro"))
# 召回率
print("micro recall score:", recall_score(target_test, result, average="micro"))
print("macro recall score:", recall_score(target_test, result, average="macro"))
```

4、数据可视化分析

为分析数据内部的分布偏差情况，使用 matplotlib.pyplot 实现可视化。分别以萼片长宽和花瓣长宽为考虑点，绘制不同种类的鸢尾花分布散点图，将结果保存为 visualize1.png 和 visualize2.png。

```
# 数据可视化分析
# 萼片长宽与标签关系散点图
x = data[:, :2]
y = target
plt.scatter(x[y == 0, 0], x[y == 0, 1], color='r', marker='o')
plt.scatter(x[y == 1, 0], x[y == 1, 1], color='g', marker='*')
plt.scatter(x[y == 2, 0], x[y == 2, 1], color='b', marker='+')
plt.xlabel('sepal length')
plt.ylabel('sepal width')
plt.savefig('visualize1.png')
plt.clf()
# 花瓣长宽与标签关系散点图
x = data[:, 2:]
y = target
plt.scatter(x[y == 0, 0], x[y == 0, 1], color='r', marker='o')
plt.scatter(x[y == 1, 0], x[y == 1, 1], color='g', marker='*')
plt.scatter(x[y == 2, 0], x[y == 2, 1], color='b', marker='+')
plt.xlabel('petal length')
plt.ylabel('petal width')
plt.savefig('visualize2.png')
plt.clf()
```

三、实验结果展示

鸢尾花数据集共 150 条，按照 7:3 划分为 105 条训练集和 45 条测试集。

测试的 micro 精确率为 97.78%，macro 准确率为 97.22%

测试的 micro 召回率为 97.78%，macro 召回率为 98.15%

整体的 SVM 分类效果较好。

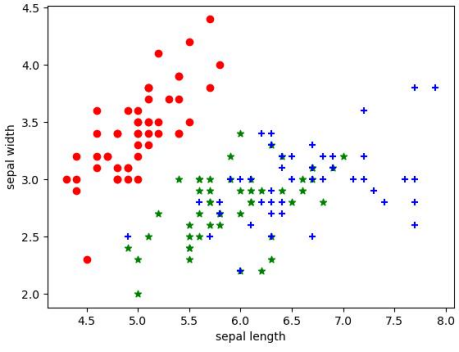
```
total size: 150 train size: 105 test_size: 45
micro precision score: 0.9777777777777777
macro precision score: 0.9722222222222222
micro recall score: 0.9777777777777777
macro recall score: 0.9814814814814815
```

四、数据集说明与可视化分析

本次 SVM 分类作业使用了经典的 Iris 鸢尾花数据集，数据集共 150 条。共 4 个特征和 1 个标签，见下表：

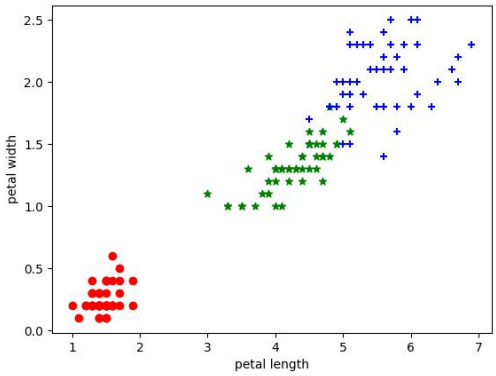
属性	类型	解释	取值
sepal length	特征	萼片长度	浮点数（厘米）
sepal width	特征	萼片宽度	浮点数（厘米）
petal length	特征	花瓣长度	浮点数（厘米）
petal width	特征	花瓣宽度	浮点数（厘米）
Species	标签	品种，共三种：Setosa、Versicolour、Virginica	0,1,2

以萼片长宽为考虑点，绘制不同种类的鸢尾花分布散点图来研究数据内部的分布偏差情况：



第一类鸢尾花萼片长度较小、宽度较大，第二类和第三类萼片宽度较第一类更小些，第二类和第三类的萼片较为相近。

以花瓣长宽为考虑点，绘制不同种类的鸢尾花分布散点图来研究数据内部的分布偏差情况：



按花瓣长宽的分布和分类应当较好，三类鸢尾花各自有集中和密集的花瓣长宽区域。特别是第一类鸢尾花，花瓣长宽的分布非常集中。