RBF 核 SVM 模型简例

南京大学 软件学院　邢骏洋*

2024 年 11 月 12 日

1　超参数之作用

1.1　C 的作用

C 是在 SVM 软间隔中引入的一个参数，它的作用是控制分类器的边界大小。相当于在对偶空间中对于 α_i 的约束条件变为了 $0 \leq \alpha_i \leq C$。通过调整 C 的大小，我们可以控制模型对于噪声数据的容忍度，或说对错误的容忍度。一般而言，C 越大，分类器对于训练数据的拟合程度就越高，容易过拟合；而 C 越小，分类器对于训练数据的拟合程度就越低，容易欠拟合。

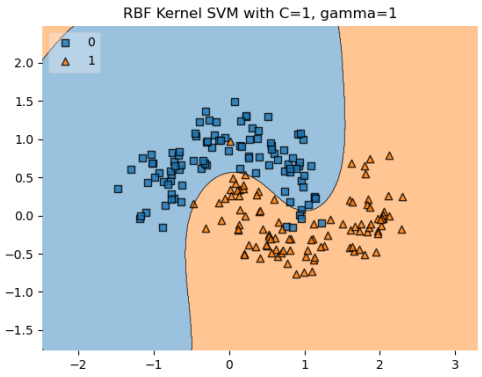

图 1: C 过高引起过拟合

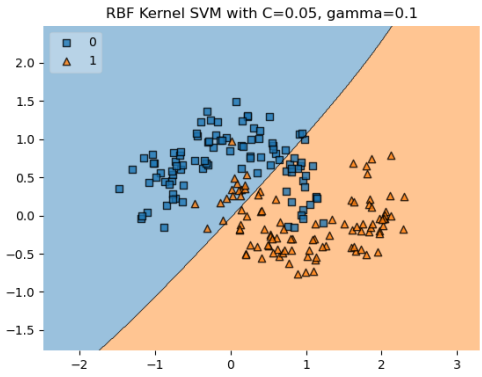

图 2: C 过低引起欠拟合

1.2　γ 的作用

RBF 核函数的定义是

$$K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2) = \langle \phi(x_i), \phi(x_j) \rangle$$

γ 越大，两个向量 x_i 和 x_j 在特征空间的距离就越小，相当于被聚集到一起了。那么，分类器就可以捕捉到更加细节的特征，在映射回原始的特征空间后，分类器的边界就会更加复杂，容易产生过拟合。反之，γ 越小，分类器的边界就会更加平滑，容易产生欠拟合。

*Software Institute, Nanjing University, Nanjing, China. E-mail: xingjunyang@smail.nju.edu.cn

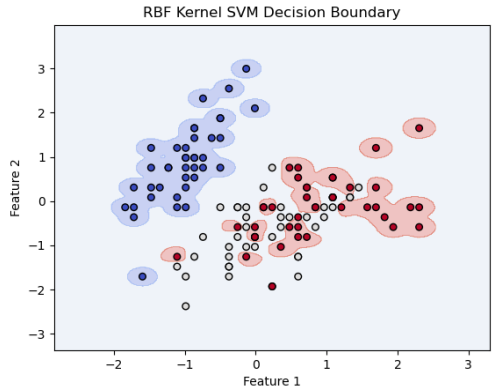

图 3: γ 过高引起过拟合

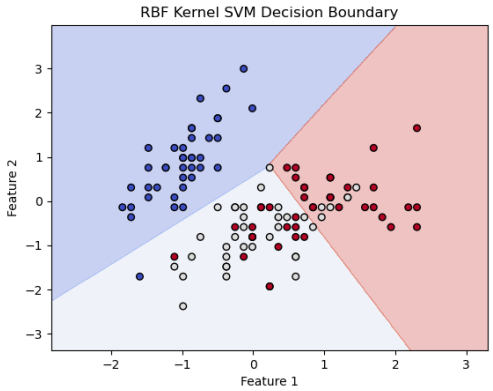

图 4: γ 过低引起欠拟合

2 步骤详析

2.1 数据准备

在本步骤中，我们加载 Iris 数据集，并选择其中的两个特征用于后续可视化。同时，我们将数据集划分为训练集和测试集，并对特征进行标准化处理，以提高 SVM 模型的收敛性和性能。

```
# 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

2.2 模型训练

在本步骤中，我们使用支持向量机（SVM）模型，分别采用线性核函数和径向基函数（RBF）核函数来训练模型。选择不同的核函数可以对比其决策边界的效果。我们设置超参数 C 和 RBF 中的 γ 来控制模型的复杂度，并使用标准化后的训练集进行拟合。

```
# 使用线性核的 SVM
linear_svm = SVC(kernel='linear', C=3, random_state=42)
linear_svm.fit(X_train, y_train)

# 使用 RBF 核的 SVM
rbf_svm = SVC(kernel='rbf', C=1, gamma=1, random_state=42)
rbf_svm.fit(X_train, y_train)
```

2.3 决策边界绘制

为了可视化模型的效果，我们定义了一个绘制决策边界的函数。通过生成数据的网格并对每个点进行分类预测，可以得到模型的决策区域。在二维平面上绘制不同类别的区域，以及训练数据点的分布。

2.4 超参数调优

使用 GridSearchCV 方法在 SVM 模型中进行超参数的调优。超参数调优能够找到最优的 C 和 γ 参数值，以提升模型的泛化能力。这里我们设置参数网格，通过三折交叉验证，并使用精确度和宏观 F1 分数进行评分，以确保模型的稳健性。

参数网格的设置为：

$$C, \gamma \in \{0.01, 0.02, \cdots, 0.99\} \cup \{0.1, 0.15, \cdots, 0.95\} \cup \{2, 12, \cdots, 92\}$$

```
# 定义参数网格
param_grid = {
    'C': [i * 0.01 for i in range(1, 100)] + [i * 0.1 for i in range(1, 100, 5)] + [i for i
    in range(2, 101, 10)],
    'gamma': [i * 0.01 for i in range(1, 100)] + [i * 0.1 for i in range(1, 100, 5)] + [i
    for i in range(2, 101, 10)],
    'kernel': ['rbf']
}

# 使用 GridSearchCV 进行网格搜索
grid = GridSearchCV(SVC(), param_grid, refit='accuracy', verbose=2, cv=3, n_jobs=-1,
    scoring=('accuracy', 'f1_macro')) # cv = 3 实现三折交叉验证
grid.fit(X_train, y_train)

# 打印最佳参数
print("Best Parameters:", grid.best_params_)
```

2.5 模型评估

在模型评估阶段，我们首先使用测试集进行预测，并计算模型的精确度、精确率、召回率和 F1 分数，以量化模型的分类效果。接着，绘制混淆矩阵，直观地显示预测与实际标签的匹配情况。最后，我们使用最佳参数的 SVM 模型绘制决策边界，以观察超参数调优后模型的决策效果。

```
# 使用最佳参数进行预测
y_pred = grid.predict(X_test)

# 绘制 RBF 核的混淆矩阵
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='coolwarm')
plt.title("RBF Kernel SVM Confusion Matrix")
plt.show()

# 绘制最佳参数的决策边界
best_svm = grid.best_estimator_
plot_decision_boundary(best_svm, X_train, y_train, "Best SVM Decision Boundary")
```

3 实验结果

3.1 代码输出结果

```
Best Parameters: {'C': 1.6, 'gamma': 0.03, 'kernel': 'rbf'}
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       0.88      0.78      0.82         9
           2       0.83      0.91      0.87        11

    accuracy                           0.90        30
   macro avg       0.90      0.90      0.90        30
weighted avg       0.90      0.90      0.90        30

Accuracy: 0.9
Precision: 0.9027777777777778
Recall: 0.8956228956228957
F1 Score: 0.8976982097186701
```

从输出结果可以看出，经过超参数调优后的 SVM 模型在测试集上的精确度为 0.9，精确率为 0.9028，召回率为 0.8956，F1 分数为 0.8977。混淆矩阵的对角线元素较大，表明模型的分类效果较好。

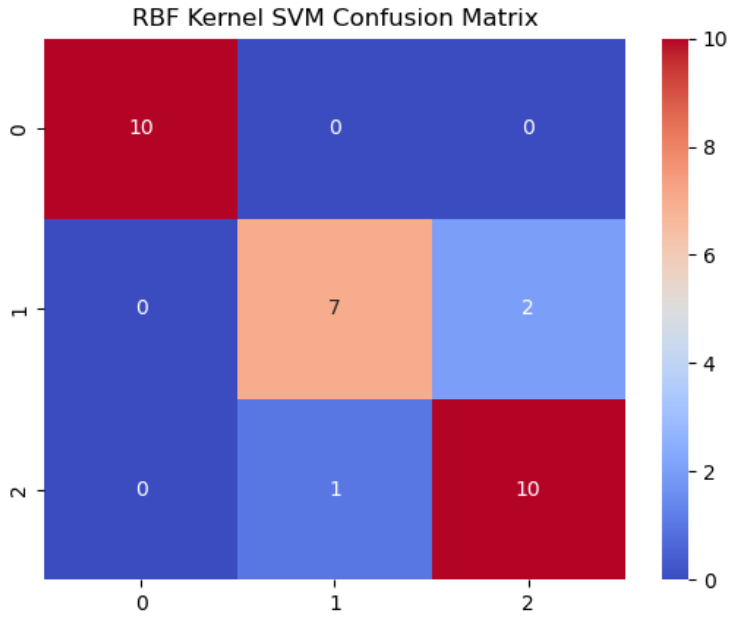

图 5: 混淆矩阵

3.2 决策边界绘制

测试性地绘制线性和 RBF 核的 SVM 模型的决策边界，可以看到线性核的决策边界是一条直线，而 RBF 核的决策边界是一条曲线。

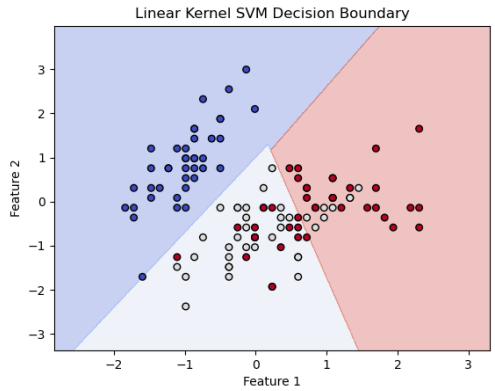

图 6: 线性 SVM

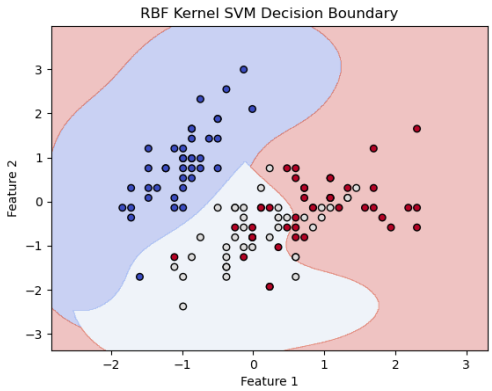

图 7: RBF 核 SVM

之后，我们使用最佳参数的 SVM 模型绘制决策边界：

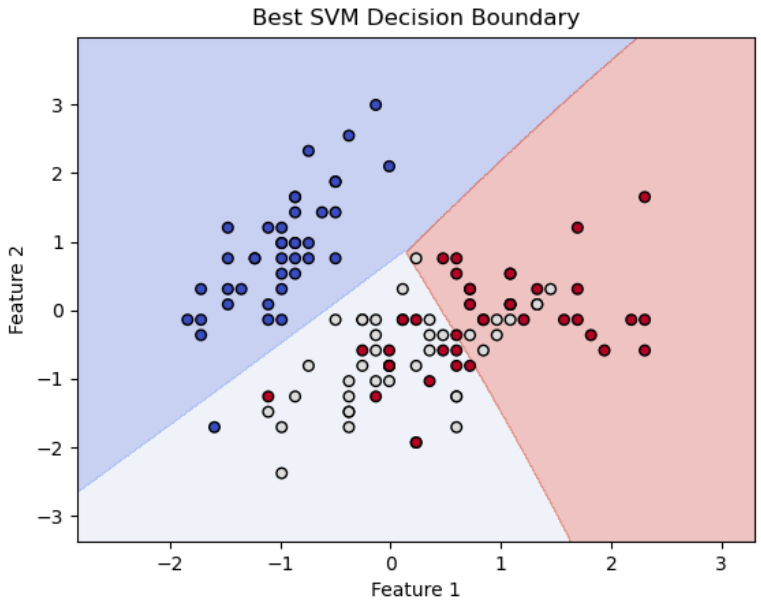

图 8: 最佳参数的 SVM 决策边界

可以看出，经过超参数调优后的 RBF 核 SVM 模型兼顾了泛化能力和分类效果，决策边界更加平滑，尽可能避免了欠拟合和过拟合，能够很好地区分不同类别的数据。

3.3 GridSearchCV 搜索过程

GridSearchCV 通过遍历参数网格，对每一组参数进行交叉验证，最终找到最佳的超参数组合。下图展示了 GridSearchCV 的搜索过程的片段：

```
[CV] END .....................C=1.6, gamma=2, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.1, gamma=0.53, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.1, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.28, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.81, kernel=rbf; total time=   0.0s
[CV] END .....................C=2.1, gamma=0.11, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.1, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.1, gamma=0.53, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.58, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=12, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.28, kernel=rbf; total time=   0.0s
[CV] END .....................C=2.1, gamma=0.11, kernel=rbf; total time=   0.0s
[CV] END .....................C=1.6, gamma=0.81, kernel=rbf; total time=   0.0s
```

由于使用了并行计算，发现参数计算的顺序并非按序的，但这可以加快计算速度。

4 线性核和 RBF 核的差异

经过本次实验，可以看出线性核只能对线性可分数据展现出较优的结果，而对于非线性可分的数据，线性核只能通过增大 C 来提高模型的性能，但这样又可能造成欠拟合。而 RBF 核在 C 的基础上又增加了 γ 这一自由度，可以和 C 进行组合，更好地拟合非线性可分的数据。因此，RBF 核的 SVM 模型在实际应用中更加灵活，能够适应更多的数据集。