

fisheriris

```
1 % 加载数据集
2 clear; clc;
3 load fisheriris;
4 X = meas;
5 y = species;
6
7 % 将类别标签转换为数值型
8 y = grp2idx(y);
9
10 % 设置重复次数
11 numRepeats = 10;
12
13 % 初始化精度数组
14 linearAccuracies = zeros(numRepeats, 1);
15 rbfAccuracies = zeros(numRepeats, 1);
16 bpAccuracies = zeros(numRepeats, 1);
17
18 % 进行多次训练和测试
19 for i = 1:numRepeats
20     % 分割数据为训练集和测试集（70% 训练，30% 测试）
21     cv = cvpartition(y, 'HoldOut', 0.3);
22     X_train = X(training(cv), :);
23     y_train = y(training(cv));
24     X_test = X(test(cv), :);
25     y_test = y(test(cv));
26
27     % 标准化特征
28     X_train = zscore(X_train);
29     X_test = zscore(X_test);
30
31     % 训练线性核SVM
32     linearSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'linear'));
33
34     % 训练高斯核SVM
35     rbfSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'rbf'));
36
37     % 训练BP神经网络
38     bpModel = feedforwardnet([10, 10]); % 增加隐藏层数量，两个隐藏层，每层10个神
经元
39     bpModel.trainParam.epochs = 200; % 增加训练迭代次数
40     bpModel.trainParam.lr = 1; % 设置学习率
41     bpModel = train(bpModel, X_train', dummyvar(y_train)'); %利用dummyvar变y
回one hot类型
42
43     % 预测线性核SVM
```

```

44 linearPredictions = predict(linearSVMModel, X_test);
45 linearAccuracies(i) = mean(linearPredictions == y_test);
46
47 % 预测高斯核SVM
48 rbfPredictions = predict(rbfSVMModel, X_test);
49 rbfAccuracies(i) = mean(rbfPredictions == y_test);
50
51 % 预测BP神经网络
52 bpPredictions = bpModel(X_test');
53 [~, bpPredictions] = max(bpPredictions, [], 1); % 将输出的概率转化为类别标签
54 bpAccuracies(i) = mean(bpPredictions' == y_test);
55 end
56
57 % 计算平均精度
58 meanLinearAccuracy = mean(linearAccuracies);
59 meanRbfAccuracy = mean(rbfAccuracies);
60 meanBpAccuracy = mean(bpAccuracies);
61
62 % 输出精度
63 fprintf('线性核SVM的平均精度: %.2f%%\n', meanLinearAccuracy * 100);
64 fprintf('高斯核SVM的平均精度: %.2f%%\n', meanRbfAccuracy * 100);
65 fprintf('BP神经网络的平均精度: %.2f%%\n', meanBpAccuracy * 100);
66
67 % 创建表格数据
68 Iteration = (1:numRepeats)';
69 LinearAccuracy = linearAccuracies * 100;
70 RBFAccuracy = rbfAccuracies * 100;
71 BPAccuracy = bpAccuracies * 100;
72
73 % 创建表格并添加平均精度行
74 resultsTable = table(Iteration, LinearAccuracy, RBFAccuracy, BPAccuracy);
75 averageRow = table(numRepeats+1, meanLinearAccuracy * 100, meanRbfAccuracy *
76     100, meanBpAccuracy * 100, 'VariableNames', {'Iteration', 'LinearAccuracy',
77     'RBFAccuracy', 'BPAccuracy'});
76 resultsTable = [resultsTable; averageRow];
77
78 % 修改表格行名称
79 resultsTable.Properties.RowNames = [cellstr(num2str((1:numRepeats)'))
80     'Average'];
81
82 % 显示表格
82 disp(resultsTable);

```

线性核svm的平均精度：97.78%
高斯核svm的平均精度：95.78%
BP神经网络的平均精度：95.33%

	Iteration	LinearAccuracy	RBFAccuracy	BPAccuracy
1	1	100	97.778	91.111
2	2	100	97.778	97.778
3	3	100	95.556	97.778
4	4	95.556	91.111	95.556
5	5	95.556	93.333	91.111
6	6	97.778	93.333	93.333
7	7	100	100	100
8	8	95.556	95.556	97.778
9	9	93.333	93.333	91.111
10	10	100	100	97.778
Average	11	97.778	95.778	95.333

Diabetes

```
1 % 加载 Diabetes 数据集
2 clear; clc;
3 load diabetes;
4 X = Feature;
5 y = Class;
6
7 % 将 one-hot 编码转换为类别标签
8 [~, y] = max(y, [], 2);
9
10 % 设置重复次数
11 numRepeats = 10;
12
13 % 初始化精度数组
14 linearAccuracies = zeros(numRepeats, 1);
15 rbfAccuracies = zeros(numRepeats, 1);
16 bpAccuracies = zeros(numRepeats, 1);
17
18 % 进行多次训练和测试
19 for i = 1:numRepeats
20     % 分割数据为训练集和测试集 (70% 训练, 30% 测试)
21     cv = cvpartition(y, 'HoldOut', 0.3);
22     X_train = X(training(cv), :);
23     y_train = y(training(cv));
24     X_test = X(test(cv), :);
25     y_test = y(test(cv));
26
27     % 标准化特征
28     X_train = zscore(X_train);
29     X_test = zscore(X_test);
30
31     % 训练线性核SVM
```

```

32     linearSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'linear'));
33
34     % 训练高斯核SVM
35     rbfSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'rbf'));
36
37
38     % 训练BP神经网络
39     bpModel = feedforwardnet([10, 10]); % 增加隐藏层数量，两个隐藏层，每层10个神
神经元
40     bpModel.trainParam.epochs = 200; % 增加训练迭代次数
41     bpModel.trainParam.lr = 1; % 设置学习率
42     bpModel = train(bpModel, X_train', dummyvar(y_train)'); %利用dummyvar变y
回one hot类型
43
44     % 预测线性核SVM
45     linearPredictions = predict(linearSVMModel, X_test);
46     linearAccuracies(i) = mean(linearPredictions == y_test);
47
48     % 预测高斯核SVM
49     rbfPredictions = predict(rbfSVMModel, X_test);
50     rbfAccuracies(i) = mean(rbfPredictions == y_test);
51
52     % 预测BP神经网络
53     bpPredictions = bpModel(X_test');
54     [~, bpPredictions] = max(bpPredictions, [], 1); % 将输出的概率转化为类别标签
55     bpAccuracies(i) = mean(bpPredictions' == y_test);
56 end
57
58 % 计算平均精度
59 meanLinearAccuracy = mean(linearAccuracies);
60 meanRbfAccuracy = mean(rbfAccuracies);
61 meanBpAccuracy = mean(bpAccuracies);
62
63 % 输出精度
64 fprintf('线性核SVM的平均精度: %.2f%%\n', meanLinearAccuracy * 100);
65 fprintf('高斯核SVM的平均精度: %.2f%%\n', meanRbfAccuracy * 100);
66 fprintf('BP神经网络的平均精度: %.2f%%\n', meanBpAccuracy * 100);
67
68 % 创建表格数据
69 Iteration = (1:numRepeats)';
70 LinearAccuracy = linearAccuracies * 100;
71 RBFAccuracy = rbfAccuracies * 100;
72 BPAccuracy = bpAccuracies * 100;
73
74 % 创建表格并添加平均精度行
75 resultsTable = table(Iteration, LinearAccuracy, RBFAccuracy, BPAccuracy);
76 averageRow = table(numRepeats+1, meanLinearAccuracy * 100, meanRbfAccuracy *
100, meanBpAccuracy * 100, 'VariableNames', {'Iteration', 'LinearAccuracy',
'RBFAccuracy', 'BPAccuracy'});

```

```

77 resultsTable = [resultsTable; averageRow];
78
79 % 修改表格行名称
80 resultsTable.Properties.RowNames = [cellstr(num2str((1:numRepeats)'))];
    'Average'];
81
82 % 显示表格
83 disp(resultsTable);

```

线性核svm的平均精度： 76.61%
 高斯核svm的平均精度： 70.00%
 BP神经网络的平均精度： 74.52%

	Iteration	LinearAccuracy	RBFAccuracy	BPAccuracy
1	1	74.783	66.087	72.609
2	2	77.391	70.87	76.522
3	3	76.087	66.522	72.609
4	4	76.087	67.391	70
5	5	75.217	70.87	76.522
6	6	77.391	71.304	74.783
7	7	75.217	73.478	73.913
8	8	78.261	73.913	78.696
9	9	78.261	68.696	76.087
10	10	77.391	70.87	73.478
Average	11	76.609	70	74.522