fisheriris

```
% 加载数据集
clear; clc;
load fisheriris;
X = meas;
y = species;

% 将类别标签转换为数值型
y = grp2idx(y);

% 设置重复次数
numRepeats = 10;

% 初始化精度数组
linearAccuracies = zeros(numRepeats, 1);
rbfAccuracies = zeros(numRepeats, 1);
bpAccuracies = zeros(numRepeats, 1);

% 进行多次训练和测试
for i = 1:numRepeats
    % 分割数据为训练集和测试集（70% 训练，30% 测试）
    cv = cvpartition(y, 'HoldOut', 0.3);
    X_train = X(training(cv), :);
    y_train = y(training(cv));
    X_test = X(test(cv), :);
    y_test = y(test(cv));

    % 标准化特征
    X_train = zscore(X_train);
    X_test = zscore(X_test);

    % 训练线性核SVM
    linearSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'linear'));

    % 训练高斯核SVM
    rbfSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'rbf'));

    % 训练BP神经网络
    bpModel = feedforwardnet([10, 10]); % 增加隐藏层数量，两个隐藏层，每层10个神
经元
    bpModel.trainParam.epochs = 200; % 增加训练迭代次数
    bpModel.trainParam.lr = 1; % 设置学习率
    bpModel = train(bpModel, X_train', dummyvar(y_train)'); %利用dummyvar变y
回one hot类型

    % 预测线性核SVM
```

```
    linearPredictions = predict(linearSVMModel, X_test);
    linearAccuracies(i) = mean(linearPredictions == y_test);

    % 预测高斯核SVM
    rbfPredictions = predict(rbfSVMModel, X_test);
    rbfAccuracies(i) = mean(rbfPredictions == y_test);

    % 预测BP神经网络
    bpPredictions = bpModel(X_test');
    [~, bpPredictions] = max(bpPredictions, [], 1); % 将输出的概率转化为类别标签
    bpAccuracies(i) = mean(bpPredictions' == y_test);
end

% 计算平均精度
meanLinearAccuracy = mean(linearAccuracies);
meanRbfAccuracy = mean(rbfAccuracies);
meanBpAccuracy = mean(bpAccuracies);

% 输出精度
fprintf('线性核SVM的平均精度: %.2f%%\n', meanLinearAccuracy * 100);
fprintf('高斯核SVM的平均精度: %.2f%%\n', meanRbfAccuracy * 100);
fprintf('BP神经网络的平均精度: %.2f%%\n', meanBpAccuracy * 100);

% 创建表格数据
Iteration = (1:numRepeats)';
LinearAccuracy = linearAccuracies * 100;
RBFAccuracy = rbfAccuracies * 100;
BPAccuracy = bpAccuracies * 100;

% 创建表格并添加平均精度行
resultsTable = table(Iteration, LinearAccuracy, RBFAccuracy, BPAccuracy);
averageRow = table(numRepeats+1, meanLinearAccuracy * 100, meanRbfAccuracy *
100, meanBpAccuracy * 100, 'VariableNames', {'Iteration', 'LinearAccuracy',
'RBFAccuracy', 'BPAccuracy'});
resultsTable = [resultsTable; averageRow];

% 修改表格行名称
resultsTable.Properties.RowNames = [cellstr(num2str((1:numRepeats)'));
'Average'];

% 显示表格
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

```matlab
% 加载 Diabetes 数据集
clear; clc;
load diabetes;
X = Feature;
y = Class;

% 将 one-hot 编码转换为类别标签
[~, y] = max(y, [], 2);

% 设置重复次数
numRepeats = 10;

% 初始化精度数组
linearAccuracies = zeros(numRepeats, 1);
rbfAccuracies = zeros(numRepeats, 1);
bpAccuracies = zeros(numRepeats, 1);

% 进行多次训练和测试
for i = 1:numRepeats
    % 分割数据为训练集和测试集（70% 训练，30% 测试）
    cv = cvpartition(y, 'HoldOut', 0.3);
    X_train = X(training(cv), :);
    y_train = y(training(cv));
    X_test = X(test(cv), :);
    y_test = y(test(cv));

    % 标准化特征
    X_train = zscore(X_train);
    X_test = zscore(X_test);

    % 训练线性核SVM
```

```
    linearSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'linear'));

    % 训练高斯核SVM
    rbfSVMModel = fitcecoc(X_train, y_train, 'Learners',
templateSVM('KernelFunction', 'rbf'));


    % 训练BP神经网络
    bpModel = feedforwardnet([10, 10]); % 增加隐藏层数量，两个隐藏层，每层10个神
经元
    bpModel.trainParam.epochs = 200; % 增加训练迭代次数
    bpModel.trainParam.lr = 1; % 设置学习率
    bpModel = train(bpModel, X_train', dummyvar(y_train)'); %利用dummyvar变y
回one hot类型

    % 预测线性核SVM
    linearPredictions = predict(linearSVMModel, X_test);
    linearAccuracies(i) = mean(linearPredictions == y_test);

    % 预测高斯核SVM
    rbfPredictions = predict(rbfSVMModel, X_test);
    rbfAccuracies(i) = mean(rbfPredictions == y_test);

    % 预测BP神经网络
    bpPredictions = bpModel(X_test');
    [~, bpPredictions] = max(bpPredictions, [], 1); % 将输出的概率转化为类别标签
    bpAccuracies(i) = mean(bpPredictions' == y_test);
end

% 计算平均精度
meanLinearAccuracy = mean(linearAccuracies);
meanRbfAccuracy = mean(rbfAccuracies);
meanBpAccuracy = mean(bpAccuracies);

% 输出精度
fprintf('线性核SVM的平均精度: %.2f%%\n', meanLinearAccuracy * 100);
fprintf('高斯核SVM的平均精度: %.2f%%\n', meanRbfAccuracy * 100);
fprintf('BP神经网络的平均精度: %.2f%%\n', meanBpAccuracy * 100);

% 创建表格数据
Iteration = (1:numRepeats)';
LinearAccuracy = linearAccuracies * 100;
RBFAccuracy = rbfAccuracies * 100;
BPAccuracy = bpAccuracies * 100;

% 创建表格并添加平均精度行
resultsTable = table(Iteration, LinearAccuracy, RBFAccuracy, BPAccuracy);
averageRow = table(numRepeats+1, meanLinearAccuracy * 100, meanRbfAccuracy *
100, meanBpAccuracy * 100, 'VariableNames', {'Iteration', 'LinearAccuracy',
'RBFAccuracy', 'BPAccuracy'});
```

```
resultsTable = [resultsTable; averageRow];

% 修改表格行名称
resultsTable.Properties.RowNames = [cellstr(num2str((1:numRepeats)')); 'Average'];

% 显示表格
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