مهدی محمدی - شماره دانشجویی: ۴۰۱۱۲۰۹۴

مجموعه آموزشی = ۷۵ نمونه تصادفی، مجموعه تست = ۷۵ نمونه باقیمانده

نتایج به‌دست‌آمده از اجرای کد متلب

Model	Training MSE	Test MSE	# of Iterations until Convergence
SVM	0.0	0.0400	…
LDA	0.0133	0.0266	…
Naive Bayes	0.0533	0.0266	…
Neural Network	…	…	…
Decision Trees	0.0133	0.0400	…

```
clc;
clear;
close all;

load fisheriris
RawData = meas(:, 1:4);
numberOfDimensions = 3;
coeff = pca(RawData);
reducedDimension = coeff(:, 1:numberOfDimensions);
reducedData1 = RawData * reducedDimension;

type = 'gaussian';
DIST = distanceMatrix(RawData);
DIST(DIST == 0) = inf;
DIST = min(DIST);
para = 5 * mean(DIST);
[reducedData2, eigVector, eigValue] = kPCA(RawData, 3, type, para);

data = [RawData, reducedData1, reducedData2];

cs = categorical(species);
ds = categories(cs);

training_x = [];
training_y = [];
testing_x = [];
testing_y = [];
for i = 1:length(ds)
    ind = find(cs == ds{i});
    % rand suffer
    ind = ind(randperm(length(ind)));
    %  50% training and 50% testing
    training_x = [training_x; data(ind(1:round(length(ind)*0.5)), :)];
    training_y = [training_y; cs(ind(1:round(length(ind)*0.5)), :)];
    testing_x = [testing_x; data(ind(1+round(length(ind)*0.5):end), :)];
    testing_y = [testing_y; cs(ind(1+round(length(ind)*0.5):end), :)];
end

% Training
clda = fitcdiscr(training_x(:, 1:10), training_y);
trainingErrlda = resubLoss(clda);
% Prediction
testClasslda = predict(clda, testing_x(:, 1:10));
testingErrlda = 1 - sum(testClasslda == testing_y) / numel(testing_y);

% Training
cnb = fitcnb(training_x(:, 1:10), training_y);
trainingErrnb = resubLoss(cnb);
% Prediction
testClassnb = predict(cnb, testing_x(:, 1:10));
testingErrnb = 1 - sum(testClassnb == testing_y) / numel(testing_y);

% Training
csvm = fitcecoc(training_x(:, 1:10), training_y);
trainingErrsvm = resubLoss(csvm);
% Prediction
testClasssvm = predict(csvm, testing_x(:, 1:10));
testingErrsvm = 1 - sum(testClasssvm == testing_y) / numel(testing_y);

ctree = fitctree(training_x(:, 1:10), training_y);
trainingErrtree = resubLoss(ctree);
% Prediction
testClasstree = predict(ctree, testing_x(:, 1:10));
testingErrtree = 1 - sum(testClasstree == testing_y) / numel(testing_y);
```