Testing PCA Models

**Table of Contents**

[Loading the Data](#MW_H_0bf4)  
[Model Calling](#MW_H_4701)

## Loading the Data

X = test\_PCA

y\_true = testResp\_PCA

## Model Calling

[yfit,scores] = PCA\_b1.predictFcn(X)

% Compute confusion matrix

confMatrix = confusionmat(y\_true, yfit);

% Display Confusion Matrix

disp('Confusion Matrix:');

disp(confMatrix);

figure(2);

heatmap(confMatrix, 'Colormap', summer); % 'summer' gives a soft green-yellow gradient

xlabel('Predicted Labels');

ylabel('True Labels');

colorbar off;

% Compute Accuracy, Precision, and Recall

numClasses = size(confMatrix, 1);

precision = diag(confMatrix) ./ sum(confMatrix, 1)'; % Column-wise precision

recall = diag(confMatrix) ./ sum(confMatrix, 2); % Row-wise recall

accuracy = sum(diag(confMatrix)) / sum(confMatrix(:)); % Overall accuracy

% Compute F1-score for each class

f1Score = 2 \* (precision .\* recall) ./ (precision + recall);

% Compute overall Precision, Recall, and F1-score

overallPrecision = mean(precision, 'omitnan');

overallRecall = mean(recall, 'omitnan');

overallF1 = mean(f1Score, 'omitnan');

% Display metrics

fprintf('Accuracy: %.2f%%\n', accuracy \* 100);

fprintf('Class-wise Precision:\n');

disp(precision);

fprintf('Class-wise Recall:\n');

disp(recall);

disp('Class-wise F1-Score:');

disp(f1Score);

% Display overall metrics

fprintf('Overall Precision: %.4f\n', overallPrecision);

fprintf('Overall Recall: %.4f\n', overallRecall);

fprintf('Overall F1-Score: %.4f\n', overallF1);

% Compute ROC curve for multi-class classification

[X, Y, T, AUC] = perfcurve(y\_true, yfit, 1); % Change '1' to the positive class label

% Plot ROC curve

figure;

plot(X, Y, 'b-', 'LineWidth', 2);

hold on;

plot([0 1], [0 1], 'k--'); % Diagonal reference line

xlabel('False Positive Rate');

ylabel('True Positive Rate');

legend(sprintf('AUC = %.2f', AUC), 'Location', 'southeast');

grid on;

hold off;