filename = 'spambaseasig.csv';

data = readmatrix(filename);

% Extract predictors + response variable

predictors = data(:, 1:end-1);

response = data(:, end);

% Splits data into training + testing

cvp = cvpartition(size(predictors, 1), 'Holdout', 0.3);

predictorsTrain = predictors(training(cvp), :);

responseTrain = response(training(cvp), :);

predictorsTest = predictors(test(cvp), :);

responseTest = response(test(cvp));

%Correlation Analysis

correlation = corr(predictorsTrain, responseTrain);

correlation = abs(correlation);

[sortedCorr, sortedIndices] = sort(correlation, 'descend');

topFeatures = sortedIndices(1:10); % Select top 10 features

% Update predictors with selected features

predictorsTrain = predictorsTrain(:, topFeatures);

predictorsTest = predictorsTest(:, topFeatures);

% Create linear SVM model

classificationSVM = fitcsvm(predictorsTrain, responseTrain, 'KernelFunction', 'linear');

% Predict response for test data using SVM

predictedOutcomeSVM = predict(classificationSVM, predictorsTest);

% Create boosted tree ensemble model

ensembleModel = fitcensemble(predictorsTrain, responseTrain, 'Method', 'AdaBoostM1');

% Predict response for test data using boosted tree

predictedOutcomeEnsemble = predict(ensembleModel, predictorsTest);

% Calculate performance metrics SVM

accuracySVM = sum(predictedOutcomeSVM == responseTest) / numel(responseTest);

precisionSVM = sum(predictedOutcomeSVM == 1 & responseTest == 1) / sum(predictedOutcomeSVM == 1);

recallSVM = sum(predictedOutcomeSVM == 1 & responseTest == 1) / sum(responseTest == 1);

f1ScoreSVM = 2 \* precisionSVM \* recallSVM / (precisionSVM + recallSVM);

% Calculate performance metrics for boosted tree

accuracyEnsemble = sum(predictedOutcomeEnsemble == responseTest) / numel(responseTest);

precisionEnsemble = sum(predictedOutcomeEnsemble == 1 & responseTest == 1) / sum(predictedOutcomeEnsemble == 1);

recallEnsemble = sum(predictedOutcomeEnsemble == 1 & responseTest == 1) / sum(responseTest == 1);

f1ScoreEnsemble = 2 \* precisionEnsemble \* recallEnsemble / (precisionEnsemble + recallEnsemble);

% Creates grouped bar chart

models = ["SVM", "Boosted Trees"];

svm\_scores = [accuracySVM, precisionSVM, recallSVM, f1ScoreSVM];

ensemble\_scores = [accuracyEnsemble, precisionEnsemble, recallEnsemble, f1ScoreEnsemble];

figure;

bar([svm\_scores; ensemble\_scores]');

ylabel("Score");

xticks(1:4);

xticklabels(["Accuracy", "Precision", "Recall", "F1 Score"]);

legend(models);

title("Performance Metrics");

ylim([0, 1]);

% Displays performance metrics for SVM

disp("SVM Performance Metrics:");

disp("------------------------");

disp("Accuracy: " + num2str(accuracySVM));

disp("Precision: " + num2str(precisionSVM));

disp("Recall: " + num2str(recallSVM));

disp("F1 Score: " + num2str(f1ScoreSVM));

disp("------------------------");

% ^ Displays for BT

disp("Boosted Trees Performance Metrics:");

disp("----------------------------------");

disp("Accuracy: " + num2str(accuracyEnsemble));

disp("Precision: " + num2str(precisionEnsemble));

disp("Recall: " + num2str(recallEnsemble));

disp("F1 Score: " + num2str(f1ScoreEnsemble));

disp("----------------------------------");

% Plots confusion matrix for SVM

figure;

confusionchart(responseTest, predictedOutcomeSVM, 'Title', 'Confusion Matrix - SVM');

% Plots confusion matrix for bt

figure;

confusionchart(responseTest, predictedOutcomeEnsemble, 'Title', 'Confusion Matrix - Boosted Tree Ensemble');