```
clear; clc; close all;
%% Question 1 %%

load fisheriris
X = meas(:,3:4);
Y = species;

rng(1); % For reproducibility



% Train a classification tree with default parameters
MdlDefault = fitctree(X,Y,'CrossVal','on');

% Plot the tree
view(MdlDefault.Trained{1},'Mode','graph')

% Cross validation classification error
classErrorDefault = kfoldLoss(MdlDefault);

% Classification Region Plot
cutPredictors   = MdlDefault.Trained{1}.CutPredictor;     % Cut Predictors
cutPoints       = MdlDefault.Trained{1}.CutPoint;         % Cut points
% Scatter plot of the original data
figure;
gscatter(X(:,1), X(:,2), species);
title('Original Data Scatter Plot and Decion Boundaries');
xlabel('Petal Length (cm)');
ylabel('Petal Width (cm)');
hold on; % Mevcut scatter plot üzerine çizim yap

for i = 1:length(cutPoints)
    if ~isnan(cutPoints(i)) % Eğer bir bölme noktası varsa
        if strcmp(cutPredictors{i}, 'x1') % Petal Length (x1)
            xline(cutPoints(i), '--k', 'LineWidth', 1.5); % Dikey çizgi
        elseif strcmp(cutPredictors{i}, 'x2') % Petal Width (x2)
            yline(cutPoints(i), '--k', 'LineWidth', 1.5); % Yatay çizgi
        end
    end
end

legend('Setosa', 'Versicolor', 'Virginica', 'Decision Boundaries');
hold off;

% Train a classification tree with maximum number of splits at 7
Mdl7 = fitctree(X,Y,'MaxNumSplits',7);

% Plot the tree
view(Mdl7.Trained{1},'Mode','graph')

% Cross validation classification error
classErrorMdl7 = kfoldLoss(Mdl7);
```

```
% Classification Region Plot
cutPredictors   = Mdl7.Trained{1}.CutPredictor;     % Cut Predictors
cutPoints       = Mdl7.Trained{1}.CutPoint;         % Cut points
% Scatter plot of the original data
figure;
gscatter(X(:,1), X(:,2), species);
title('Original Data Scatter Plot and Decion Boundaries');
xlabel('Petal Length (cm)');
ylabel('Petal Width (cm)');
hold on; % Mevcut scatter plot üzerine çizim yap

for i = 1:length(cutPoints)
    if ~isnan(cutPoints(i)) % Eğer bir bölme noktası varsa
        if strcmp(cutPredictors{i}, 'x1') % Petal Length (x1)
            xline(cutPoints(i), '--k', 'LineWidth', 1.5); % Dikey çizgi
        elseif strcmp(cutPredictors{i}, 'x2') % Petal Width (x2)
            yline(cutPoints(i), '--k', 'LineWidth', 1.5); % Yatay çizgi
        end
    end
end

legend('Setosa', 'Versicolor', 'Virginica', 'Decision Boundaries');
hold off;


% Train a classification tree with Entropy SplitCriterion
MdlSplit = fitctree(X,Y,'SplitCriterion','deviance','CrossVal','on');

% Plot the tree
view(MdlSplit.Trained{1},'Mode','graph')

% Cross validation classification error
classErrorMdlSplit = kfoldLoss(MdlSplit);

% Classification Region Plot
cutPredictors   = MdlSplit.Trained{1}.CutPredictor;     % Cut Predictors
cutPoints       = MdlSplit.Trained{1}.CutPoint;         % Cut points
% Scatter plot of the original data
figure;
gscatter(X(:,1), X(:,2), species);
title('Original Data Scatter Plot and Decion Boundaries');
xlabel('Petal Length (cm)');
ylabel('Petal Width (cm)');
hold on; % Mevcut scatter plot üzerine çizim yap

for i = 1:length(cutPoints)
    if ~isnan(cutPoints(i)) % Eğer bir bölme noktası varsa
        if strcmp(cutPredictors{i}, 'x1') % Petal Length (x1)
            xline(cutPoints(i), '--k', 'LineWidth', 1.5); % Dikey çizgi
        elseif strcmp(cutPredictors{i}, 'x2') % Petal Width (x2)
            yline(cutPoints(i), '--k', 'LineWidth', 1.5); % Yatay çizgi
        end
    end
```

```
end

legend('Setosa', 'Versicolor', 'Virginica', 'Decision Boundaries');
hold off;
clc; clear; close all;
%% Question 2 %%
rng(100);
load fisheriris
X = meas(:,3:4);
Y = species;

% Partition the data into two halves
cv = cvpartition(Y, 'Holdout', 0.5); % Split data into training and testing sets

idxX1 = training(cv);                % Indices for X1
idxX2 = test(cv);                    % Indices for X2
idxY1 = training(cv);                % Indices for Y1
idxY2 = test(cv);                    % Indices for Y2

X1 = X(idxX1, :);
Y1 = Y(idxY1);
X2 = X(idxX2, :);
Y2 = Y(idxY2);

weak_learner = templateTree("MaxNumSplits",1);
% First half of the observation
Mdl = fitcensemble(X1,Y1,'Method','AdaBoostM2','NumLearningCycles',25,'Learners',↙
weak_learner);
% Plot the tree
view(Mdl.Trained{1},'Mode','graph')

% Predict the labels from second half of the data using ensemble model
PredictYensemble = predict(Mdl,X2);

% Predict the labels from second half of the data using first tree
PredictYfirsttree = predict(Mdl.Trained{1},X2);

% Calculate classification accuracy for both
accuracyEnsemble    = 0;
accuracyFirstTree   = 0;
for i = 1:numel(Y2)
    accuracyEnsemble  = accuracyEnsemble  + strcmp(PredictYensemble{i},Y2{i});
    accuracyFirstTree = accuracyFirstTree + strcmp(PredictYfirsttree{i}, Y2{i});
end
accuracyEnsemble = accuracyEnsemble / numel(Y2);
accuracyFirstTree = accuracyFirstTree / numel(Y2);
% Display classification accuracies
disp(['Accuracy of AdaBoost ensemble: ', num2str(accuracyEnsemble)]);
disp(['Accuracy of the first tree: ', num2str(accuracyFirstTree)]);

% LearnRate Parameters

lr = 0.01:0.01:1;
```

```
acc = zeros(size(lr));
for i = 1:length(lr)
    weak_learner = templateTree("MaxNumSplits",1);
    Mdl = fitcensemble(X1,Y1,'Method','AdaBoostM2','NumLearningCycles',↙
25,'Learners',weak_learner,"LearnRate",lr(i));
    Yensemble = predict(Mdl,X2);
    Yfirsttree = predict(Mdl.Trained{1},X2);

    % Calculate classification accuracy
    accEnsemble    = 0;
    for ii = 1:numel(Y2)
        accEnsemble  = accEnsemble  + strcmp(Yensemble{ii},Y2{ii});
    end
    accEnsemble = accEnsemble / numel(Y2);
    acc(i) = accEnsemble;
end

figure;
plot(lr,acc)
xlabel("Learning Rate")
ylabel("Accuracy")
title("Accuracy vs. Learning Rate")
clc; clear; close all;
%% Question 3 %%
rng(100);
load fisheriris
X = meas(:,3:4);
Y = species;

% Partition the data into two halves
cv = cvpartition(Y, 'Holdout', 0.5); % Split data into training and testing sets

idxX1 = training(cv);                % Indices for X1
idxX2 = test(cv);                    % Indices for X2
idxY1 = training(cv);                % Indices for Y1
idxY2 = test(cv);                    % Indices for Y2

X1 = X(idxX1, :);
Y1 = Y(idxY1);
X2 = X(idxX2, :);
Y2 = Y(idxY2);
% rng(100);

Mdl = TreeBagger(25,X1,Y1,'SampleWithReplacement','on','OOBPrediction',"on","↙
Method","classification");

view(Mdl.Trees{1},'Mode','graph')
%Prediction
% Predict the labels from second half of the data using ensemble model
PredictYensemble = predict(Mdl,X2);

% Predict the labels from second half of the data using first tree
PredictYfirsttree = predict(Mdl.Trees{1},X2);
```

```
% Calculate classification accuracy for both
accuracyEnsemble    = 0;
accuracyFirstTree   = 0;
for i = 1:numel(Y2)
    accuracyEnsemble  = accuracyEnsemble  + strcmp(PredictYensemble{i},Y2{i});
    accuracyFirstTree = accuracyFirstTree + strcmp(PredictYfirsttree{i}, Y2{i});
end
accuracyEnsemble = accuracyEnsemble / numel(Y2);
accuracyFirstTree = accuracyFirstTree / numel(Y2);
% Display classification accuracies
disp(['Accuracy of AdaBoost ensemble: ', num2str(accuracyEnsemble)]);
disp(['Accuracy of the first tree: ', num2str(accuracyFirstTree)]);

plot(oobError(Mdl))
xlabel("Number of Grown Trees")
ylabel("Out-of-Bag Classification Error")
clc; clear; close all;
%% Question 4 Jackknife sampling %%

% First compute the sample correlation on the data
load lawdata
rhohat = corr(lsat,gpa);

plot(lsat,gpa,'+')
lsline
xlabel('LSAT Scores')    % Label x-axis
ylabel('GPA Scores')     % Label y-axis
title('LSAT vs GPA')     % Plot title
grid on                  % Add grid for better visualization

% Next compute the correlations for jackknife samples, and compute their mean
rng default;  % For reproducibility
jackrho = jackknife(@corr,lsat,gpa);
meanrho = mean(jackrho);

% Compute an estimate of the bias
n = length(lsat);
biasrho = (n-1) * (meanrho-rhohat);

% histogram(jackrho,30,'FaceColor',[.8 .8 1])

% Median Calculation for LSAT
medhat_lsat = median(lsat);
jackmed_lsat = jackknife(@median,lsat);

biasmed_lsat = (n-1) * (mean(jackmed_lsat)- (medhat_lsat));

% Median Calculation for GPA
medhat_gpa = median(gpa);
jackmed_gpa = jackknife(@median,gpa);
```

```
biasmed_gpa = (n-1) * (mean(jackmed_gpa)- mean(medhat_gpa));

disp(['Sample Mean : ', num2str((rhohat))]);
disp(['Mean Jackknife Correlation Coefficient: ', num2str(meanrho)]);
disp(['Jackknife Estimate of Bias: ', num2str(biasrho)]);


disp(['Sample Median LSAT: ', num2str((medhat_lsat))]);
disp(['Jackknife Average Median LSAT: ', num2str(mean(jackmed_lsat))]);
disp(['Jackknife Bias LSAT: ', num2str(biasmed_lsat)]);

disp(['Sample Median GPA: ', num2str(medhat_gpa)]);
disp(['Jackknife Average Median GPA: ', num2str(mean(jackmed_gpa))]);
disp(['Jackknife Bias GPA: ', num2str(biasmed_gpa)]);
```