EE583 Pattern Recognition HW5

Kutay Uğurlu 2232841

January 2, 2022

1 Question 1

3 different experiments are conducted. In each experiment, ten models are individually used to calculate the accuracy, or equivalently misclassification loss.

1.1 Default settings

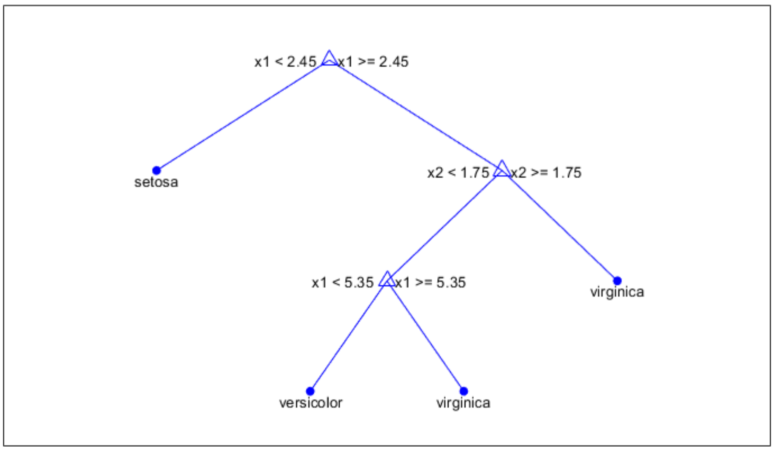


Figure 1: Visualization of the first tree

This 3 split tree resulted in a loss of 0.0267.

1.2 Maximum number of Splits Restriction

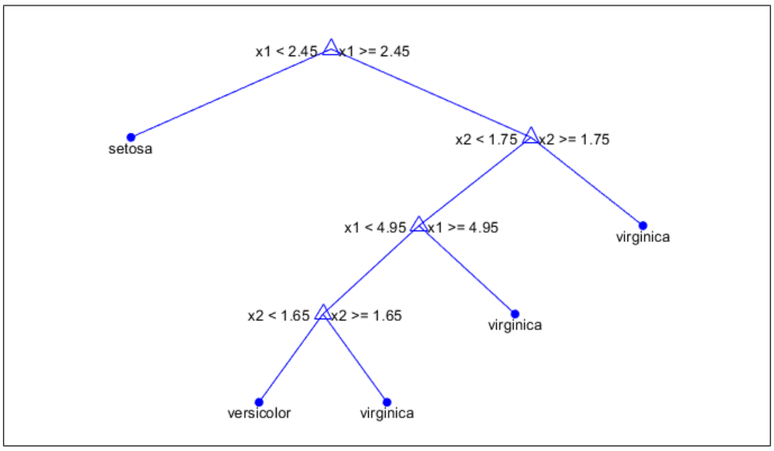


Figure 2: Visualization of the second tree

For this experiment the resultant number of splits was higher than the first one. Therefore, it resulted in less misclassification loss of 0.0240.

1.3 Maximum number of Splits Restriction

Changing the split criterion from Gini's diversity index to deviance resulted in the improvement of the accuracy.

2 Question 2

2.1 Single tree vs Ensemble

To observe the performance difference between a single tree and the ensemble better, the number of splits were set to 1 to decrease the classification capability of the single tree. Otherwise, even the first week learner is able to classify the data accurate enough.

$$Accuracy_{first} = 0.64 \qquad Accuracy_{ensemble} = 0.95$$

The results reflect our expectations on the performance of the ensemble trained with Adaboost being better.

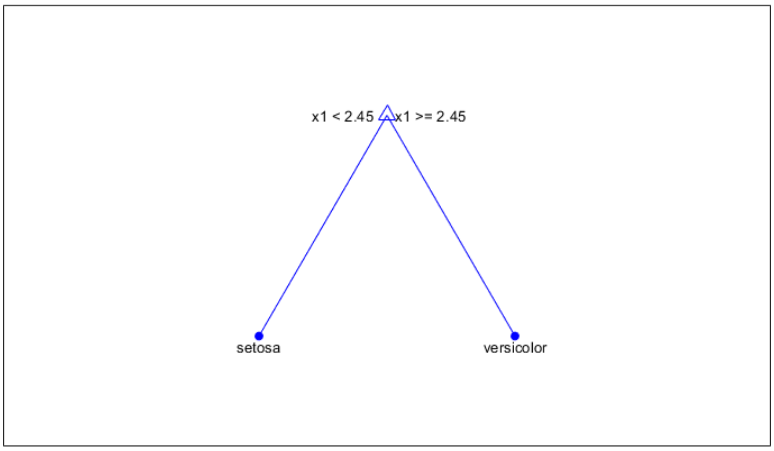


Figure 3: Visualization of the first tree

2.2 Learning Rate

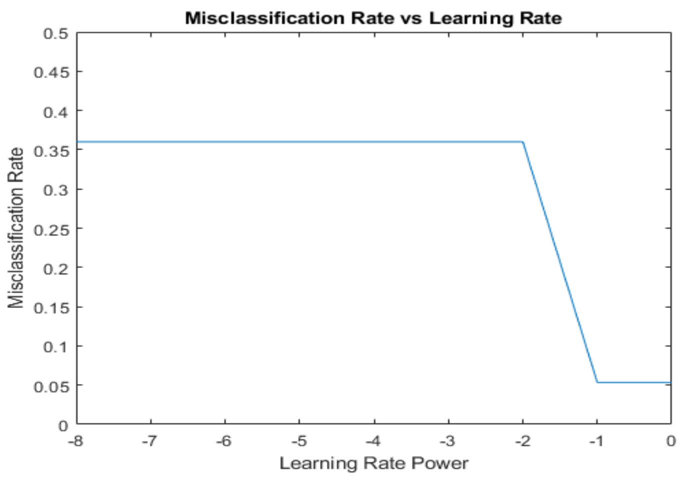


Figure 4: Learning Rate vs Misclassification

Again, the maximum number of splits is fixed among experiments, which can be seen in 5.2. Larger learning rates resulted ensemble to achieve higher accuracy, since the informative samples have more weights in the training process.

3 Question 3

3.1 Learning Rate

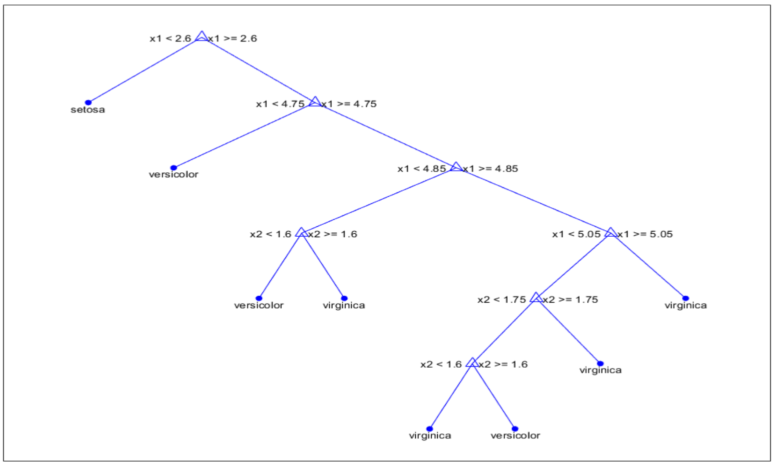
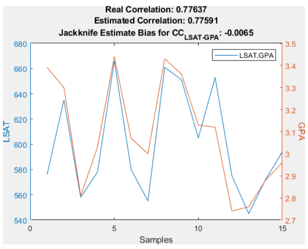


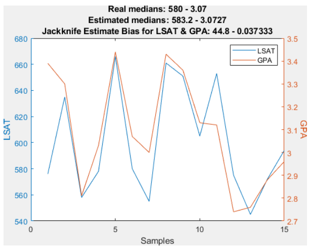
Figure 5: First Tree

The accuracy of the first tree is calculated as 0.9867 whereas the forest's is 1. We see that, with the help of the 24 remaining tree models in the treebagger object, the model was able to classify the samples correctly that are misclassified by the first tree. Hence, the classification accuracy improved.

4 Question 4



(a) Correlation Estimate



(b) Median Estimate

Figure 6: Jackknife estimates

The variables are plotted with respect to different axes in both Figure 6a and 6b. The estimates and the estimated bias are calculated and printed in the title. The resultant biases are respectively low when the variance of the relative variables are taken into account.

5 APPENDIX

The code given in this section is shared @.

5.1 Q1

```
%%
clear;  clc;
chdir('..')
addpath('export_fig')
chdir('HW5')
rng(101)
%%
load fisheriris.mat
feats = meas(:,3:4);
Y = species;
%%
tree_model = fitctree(feats,species,'CrossVal','on');
view(tree_model.Trained{1},'Mode','graph')
Ls = [];
for i = 1:10
    model = tree_model.Trained{i};
    preds = predict(model,feats);
    confusion_matrix = confusionmat(species,preds);
    accuracy = sum(diag(confusion_matrix))/sum(sum(confusion_matrix))
        ;
    loss = 1 - accuracy;
    Ls(end+1) = loss;
end
mean_loss_default = mean(Ls);

%%
tree_model = fitctree(feats,species,'CrossVal','on','MaxNumSplits',7)
    ;
view(tree_model.Trained{1},'Mode','graph')
Ls = [];
for i = 1:10
    model = tree_model.Trained{i};
    preds = predict(model,feats);
    confusion_matrix = confusionmat(species,preds);
    accuracy = sum(diag(confusion_matrix))/sum(sum(confusion_matrix))
        ;
    loss = 1 - accuracy;
    Ls(end+1) = loss;
end
mean_loss_restricted_splits = mean(Ls);
%%
tree_model = fitctree(feats,species,'CrossVal','on','SplitCriterion',
    'deviance');
view(tree_model.Trained{1},'Mode','graph')
Ls = [];
for i = 1:10
```

```
    model = tree_model.Trained{i};
    preds = predict(model,feats);
    confusion_matrix = confusionmat(species,preds);
    accuracy = sum(diag(confusion_matrix))/sum(sum(confusion_matrix))
        ;
    loss = 1 - accuracy;
    Ls(end+1) = loss;
end
mean_loss_split_criterion = mean(Ls);
%%
figHandles = findall(0,'Type','figure');

for i = 1:numel(figHandles)
    export_fig(['Q1_',num2str(i)], '-png', figHandles(i), '-append')
end

hTree=findall(0,'Tag','tree viewer');
% close(hTree)

mean_loss_default
mean_loss_restricted_splits
mean_loss_split_criterion
```

5.2 Q2

```
clc, clear;
load fisheriris.mat
feats = meas(:,3:4);
Y = species;
rng(101)

% Cross varidation (train: 50%, test: 50%)
cv = cvpartition(size(feats,1),'HoldOut',0.5);
idx = cv.test;

% Separate to training and test data
feats_Train = feats(~idx,:);
feats_Test = feats(idx,:);
Y_Train = Y(~idx);
Y_Test = Y(idx);

% To amplify the difference of the classification success, the number of
% splits are restricted for a single tree to also highlight the adaboost
% success
t = templateTree('MaxNumSplits',1);
Mdl = fitcensemble(feats_Train,Y_Train,'Method','AdaBoostM2', ...
    'Learners',t,'NumLearningCycles',25);
view(Mdl.Trained{1},'Mode','graph')
preds = predict(Mdl.Trained{1},feats_Test);
confusionmatrix = confusionmat(Y_Test,preds);
first_tree_accuracy = sum(diag(confusionmatrix))/sum(sum(confusionmatrix))
preds = predict(Mdl,feats_Test);
confusionmatrix = confusionmat(Y_Test,preds);
ensemble_tree_accuracy = sum(diag(confusionmatrix))/sum(sum(confusionmatrix))


accs = [];
for lr = 10.^[-8:0]
    t = templateTree('MaxNumSplits',3);
    Mdl = fitcensemble(feats_Train,Y_Train,'Method','AdaBoostM2', ...
    'Learners',t,'NumLearningCycles',25, 'LearnRate',lr);
    preds = predict(Mdl,feats_Test);
    confusionmatrix = confusionmat(Y_Test,preds);
    model_accuracy = sum(diag(confusionmatrix))/sum(sum(confusionmatrix));
    accs(end+1) = model_accuracy;
end

figure
plot(-8:0,1-accs)
```

```
ylim([0 0.5])
title('Misclassification Rate vs Learning Rate')
ylabel('Misclassification Rate')
xlabel('Learning Rate Power')
```

5.3 Q3

```
clc, clear;
load fisheriris.mat
feats = meas(:,3:4);
Y = species;
rng(101)

% Cross varidation (train: 50%, test: 50%)
cv = cvpartition(size(feats,1),'HoldOut',0.5);
idx = cv.test;

% Separate to training and test data
feats_Train = feats(~idx,:);
feats_Test = feats(idx,:);
Y_Train = Y(~idx);
Y_Test = Y(idx);

Mdl = TreeBagger(25,feats,Y,'OOBPrediction','On',...
    'Method','classification','SampleWithReplacement','on');


view(Mdl.Trees{1},'Mode','graph')
preds = predict(Mdl.Trees{1},feats_Test);
confusionmatrix = confusionmat(Y_Test,preds);
first_tree_accuracy = sum(diag(confusionmatrix))/sum(sum(
    confusionmatrix))
preds = predict(Mdl,feats_Test);
confusionmatrix = confusionmat(Y_Test,preds);
ensemble_tree_accuracy = sum(diag(confusionmatrix))/sum(sum(
    confusionmatrix))
```

5.4 Q4

```matlab
load lawdata
rhohat = corr(lsat,gpa);

%%
rng default; % For reproducibility
jackrho = jackknife(@corr,lsat,gpa);
meanrho = mean(jackrho);
yyaxis left
plot(lsat)
ylabel('LSAT')
hold on
yyaxis right
plot(gpa)
h = ylabel('GPA','Rotation',270);
xlabel('Samples')
h.Position(1) = 16.5; % change horizontal position of ylabel
legend('LSAT,GPA')
n = length(lsat);
biasrho = round((n-1) * (meanrho-rhohat),4);
title({['Real Correlation: ',num2str(rhohat)],['Estimated Correlation: ',num2str(meanrho)]},['Jackknife Estimate Bias for CC_{LSAT-GPA}: ',num2str(biasrho)],'FontWeight','bold')



%%
rng default; % For reproducibility
jackmed = jackknife(@median,lsat);
meanmed_lsat = mean(jackmed);
n = length(lsat);
bias_lsat_median = (n-1) * (meanmed_lsat-median(lsat))

jackmed = jackknife(@median,gpa);
meanmed_gpa = mean(jackmed);
n = length(gpa);
bias_gpa_median = (n-1) * (meanmed_gpa-median(gpa))

figure
yyaxis left
hp1 = plot(lsat);
ylabel('LSAT')
hold on
yyaxis right
hp2 = plot(gpa);
legend([hp1,hp2],'LSAT','GPA')
xlabel('Samples')
h = ylabel('GPA','Rotation',270);
h.Position(1) = 16.5; % change horizontal position of ylabel
```

```
title({['Real medians: ',num2str(median(lsat)),' - ',num2str(median(gpa))],['Estimated medians: ',num2str(meanmed_lsat),' - ',num2str(meanmed_gpa)],['Jackknife Estimate Bias for LSAT & GPA: ',num2str(bias_lsat_median),' - ',num2str(bias_gpa_median)]})



figHandles = findall(0,'Type','figure');

for i = 1:numel(figHandles)
    export_fig(['Q4_',num2str(i)], '-png', figHandles(i), '-append')
end
close all
```