

机器学习第五周作业

樊泽羲 2200010816

Q 1:给出随机森林的算法框架

A 1: 我们用两个算法实现, 学习好的森林称为 f 。在输入特定样本时, 在 f 的所有决策树中进行多数表决即可生成该样本的预测结果

Algorithm 1. 随机选择特征, 训练一棵决策树

输入: 训练数据集 D , 特征集 F , 阈值 ϵ

输出: 在 F 中随机一部分特征上训练好的决策树 T

1. 如果 D 已被纯化, 或者 $F = \emptyset$, 则通过多数表决生成 T 的类标签 C_k , 返回 T
2. 否则, 在 F 中随机选择一小部分, 记为 \hat{F} , 对当前数据集 D 计算 \hat{F} 中各个特征的信息增益率, 选择增益率最大特征 A
3. 如果 A 的增益率小于 ϵ , 则通过多数表决生成 T 的类标签 C_k , 返回 T
4. 否则, 对 A 的每个取值 a_j , 将 D 分割为若干子集 D_i , 将 D_i 中实例数最大的类作为标记, 构建子节点, 由结点及其子节点构成树 T , 返回 T
5. 对于第 i 个子节点, 以 D_i 为训练集, 以 $F - A$ 作为特征集, 递归调用1-4, 得到 T_i , 返回 T_i

Algorithm 2. 利用随机训练的决策树构建随机森林

输入: 训练数据集 $S = \{(x_j, y_j)\}_{j=1}^n$, 特征集 F , f 中所含的决策树个数 N

输出: 训练好的随机森林 f

1. f 初始化为 \emptyset
2. 从 S 中用自助法采样 N 次, 产生训练数据 $S_k, 1 \leq k \leq N$
3. 对于每个 S_k , 调用Algorithm 1训练决策树 T^k , 并将 T^k 添加进 f
4. 返回 f

Q 2:随机森林和Bagging算法的主要思想比较

A 2:

共同点:

1. 从目的上看: 都想通过重采样方法增加决策树的泛化能力。决策树(比如CART)通过贪婪搜索处理NP-hard问题, 在预测中一般存在显著的过拟合及方差, 两种方法本质上都是在通过集成差异较大的树减少这个偏差
2. 从训练集的处理上看: 都对训练数据集进行了自助法采样
3. 从结果生成来看: 都通过多数表决生成分类结果, 通过加权平均生成回归问题的结果

不同点:

1. 从特征选择上看: bagging在每一步中都选择所有特征进行训练, 随机森林在每一步中选择不同特征进行训练
2. 从适用范围上看: bagging方法可以广泛用于各种基学习器的集成学习, 但是随机森林一般仅能用于决策树的集成

3. 从优势劣势上看：相较随机森林，bagging充分利用了所有特征，当estimator高度不稳定时更加奏效，因此常常用于提升特定算法的稳定性。相较bagging,random forest进行了两次bootstrap，因此过拟合的风险更小，方差一般也更小，更适合用作预测模型

Q 3:随机森林和Bagging算法的实验结果比较

数据预处理

本实验使用了四个公开数据集：

Breast Cancer Wisconsin (Diagnostic) Data Set: ¹

Samples: 569

Features: 30

Mission: Classification

Classes: Benign, Malignant

Wine Dataset: ²

Samples: 178

Features: 13

Mission: Classification

Classes: 3 varieties of wine

Student Performance Dataset: ³

Samples: 649

Features: 32

Mission: Regression

Computer Hardware Dataset: ⁴

Samples: 209

Features: 7

Mission: Regression

对于乳腺癌和计算机硬件数据集，没有缺失值。葡萄酒数据集有一个缺失数据的样本被删除

采用5-fold交叉验证，并且对于分类任务，在5个不同数据集中尽量保持labels的各个类别比例保持均匀

模型建立

Random Forest:

使用matlab实现随机森林分类器。关键超参数如下：

Breast Cancer Wisconsin (Diagnostic) Data Set:

集成了 100 个装袋决策树:

```
Training X: [569x30]
Training Y: [569x1]
Method: classification
NumPredictors: 30
NumPredictorsToSample: 6
MinLeafSize: 5
InBagFraction: 1
SampleWithReplacement: 1
ComputeOOBPrediction: 1
ComputeOOBPredictorImportance: 1
Proximity: []
ClassNames: 'B' 'M'
```

Wine Dataset:

集成了 100 个装袋决策树:

```
Training X: [178x13]
Training Y: [178x1]
Method: classification
NumPredictors: 13
NumPredictorsToSample: 4
MinLeafSize: 5
InBagFraction: 1
SampleWithReplacement: 1
ComputeOOBPrediction: 1
ComputeOOBPredictorImportance: 1
Proximity: []
ClassNames: '1' '2' '3'
```

Student Performance Dataset:

集成了 100 个装袋决策树:

```
Training X: [395x32]
Training Y: [395x1]
Method: regression
NumPredictors: 32
NumPredictorsToSample: 11
MinLeafSize: 5
InBagFraction: 1
SampleWithReplacement: 1
ComputeOOBPrediction: 1
ComputeOOBPredictorImportance: 1
Proximity: []
```

Computer Hardware Dataset:

集成了 100 个装袋决策树:

```
Training X: [209x9]
Training Y: [209x1]
Method: regression
NumPredictors: 9
NumPredictorsToSample: 3
MinLeafSize: 5
InBagFraction: 1
SampleWithReplacement: 1
ComputeOOBPrediction: 1
ComputeOOBPredictorImportance: 1
Proximity: []
```

使用matlab的Statistics and Machine Learning Toolbox默认调优。对于每个数据集，最优值是不同的

Bagging:

使用matlab实现bagging分类器。我们使用相同的基学习器决策树和相同的默认调参策略来进行公平比较:

Breast Cancer Wisconsin (Diagnostic) Data Set:

```
CrossValidatedModel: 'Bag'
PredictorNames: {1x30 cell}
ResponseName: 'Diagnosis'
NumObservations: 569
KFold: 5
Partition: [1x1 cvpartition]
NumTrainedPerFold: [100 100 100 100 100]
ClassNames: [B M]
ScoreTransform: 'none'
```

Wine Dataset:

```

ClassificationPartitionedEnsemble
  CrossValidatedModel: 'Bag'
    PredictorNames: {1×13 cell}
    ResponseName: 'class'
    NumObservations: 178
    KFold: 5
    Partition: [1×1 cvpartition]
    NumTrainedPerFold: [100 100 100 100 100]
    ClassNames: [1 2 3]
    ScoreTransform: 'none'

```

Student Performance Dataset:

```

RegressionPartitionedEnsemble
  CrossValidatedModel: 'Bag'
    PredictorNames: {1×32 cell}
    CategoricalPredictors: [1 2 4 6 9 10 11 12 16 17 18 19 20 21 22 23]
    ResponseName: 'G3'
    NumObservations: 395
    KFold: 5
    Partition: [1×1 cvpartition]
    NumTrainedPerFold: [100 100 100 100 100]
    ResponseTransform: 'none'

```

Computer Hardware Dataset:

```

RegressionPartitionedEnsemble
  CrossValidatedModel: 'Bag'
    PredictorNames: {'vendorName' 'modelName' 'MYCT' 'MMIN' 'MMAX' 'CACH' 'CHMIN' 'CHMAX' 'PRP'}
    CategoricalPredictors: [1 2]
    ResponseName: 'ERP'
    NumObservations: 209
    KFold: 5
    Partition: [1×1 cvpartition]
    NumTrainedPerFold: [100 100 100 100 100]
    ResponseTransform: 'none'

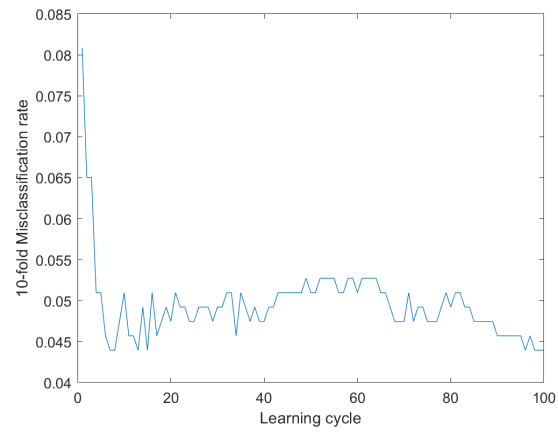
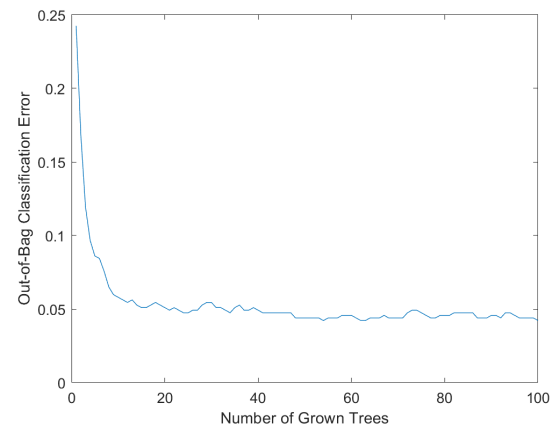
```

random forest和bagging模型均在预处理后的训练数据上进行训练。测试集用于报告每个模型在以前未见过的数据上的最终性能

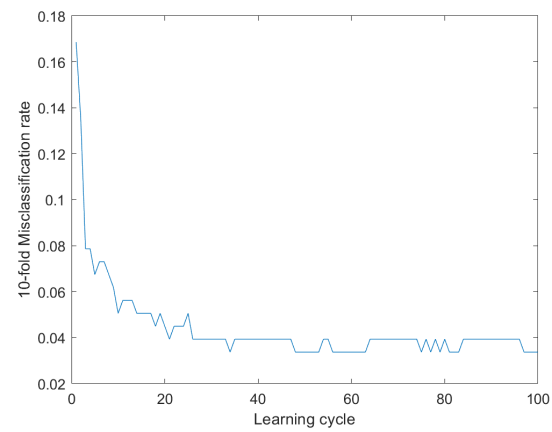
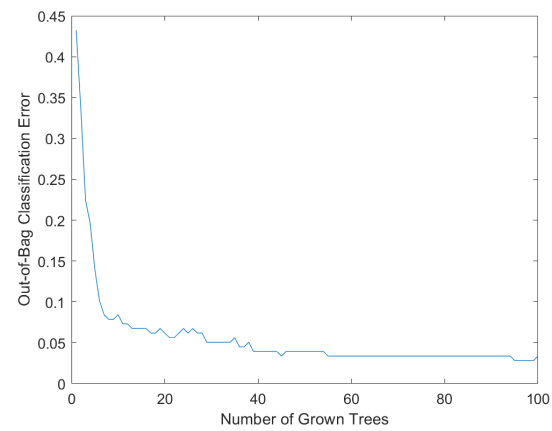
结果分析

分别采用out-of-bag error和mse评估两种模型在classification和regression中的表现，结果如下所示，其中上图图代表RF的表现，下图图代表Bagging的表现（横纵坐标的标注可能不同，但实际检验的是相同指标）：

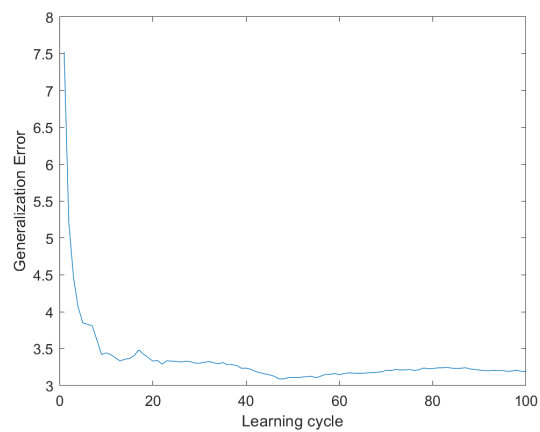
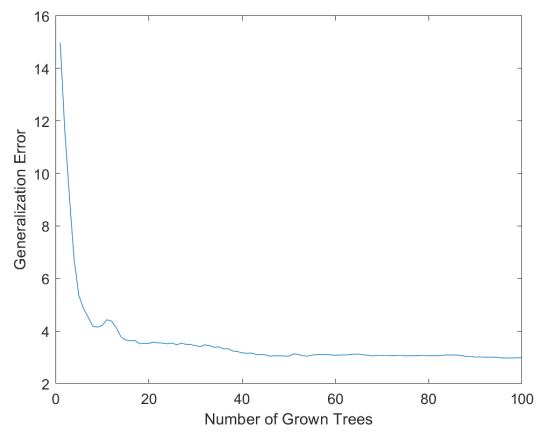
Breast Cancer Wisconsin (Diagnostic) Data Set:



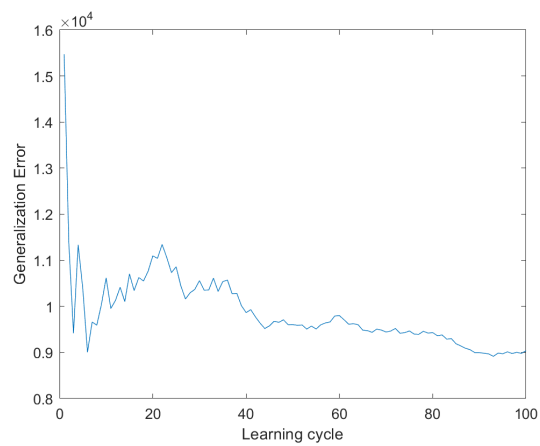
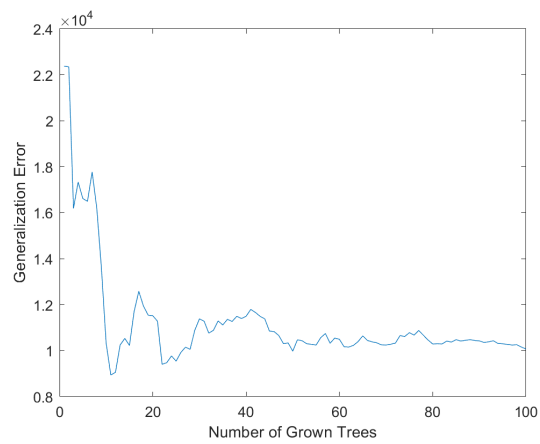
Wine Dataset:



Student Performance Dataset:



Computer Hardware Dataset:



从这些结果中可以观察到一些关键趋势

随机森林在所有四个数据集上的表现都劣于bagging，每种情况下的泛化误差均略高。这表明，通过特征采样在随机森林中引入的额外随机性，可能不利于它在这些分类任务上更好地泛化

两种算法都在特征最少的葡萄酒数据集上表现最好。像计算机硬件和乳腺癌这样的复杂数据集被证明更加困难，表现出更高的损失。特征之间较少的关系可能更有利于基于树的集成模型

学生成绩数据集中，随机森林和bagging之间的成绩差异最小。这表明此预测任务更容易，引入随机性导致的模型不稳定影响更小

总之，在这些数据集上，随机森林的泛化能力低于bagging树。在每次划分时进行更多的随机特征选择似乎有助于防止过拟合，但是可能也造成了偏差的增大。然而，二者之间的差异整体很小，前者对于分类任务也很有效，而且计算量更小。未来的工作可以在更多多样化的问题上比较其他集成方法

结论

本文在4个基准数据集上比较了随机森林和bagging分类器的性能。主要发现如下：

随机森林在所有四个数据集上的表现都劣于bagging，每种情况下的泛化误差均略高。这表明，通过特征采样在随机森林中引入的额外随机性，可能不利于它在这些分类任务上更好地泛化

具有较少特征的葡萄酒数据集在两种模型中均表现出最佳性能，突出了集成树在具有较少特征关系的简单问题上的有效性

像乳腺癌和计算机硬件这样的复杂数据集表现出更高的损失，表明用这些浅层模型对高度多元数据建模面临较大挑战

学生数据集在随机森林和bagging性能上的差异最小，这表明更简单的分类任务引入随机性导致的模型不稳定影响更小

这项研究的一些局限性为未来的工作提供了途径。例如，只在四个公开数据集比对了两个模型，在更广泛的问题上比较这些模型可能会产生不同的结果。此外，本文没有探讨深度学习和其他先进的集成方法，但它们是很有价值的对照组

综上所述，在所分析的分类问题上，随机森林比bagging tree稍有劣势。然而，两种浅层集成都被证明是有效的学习器，前者也在这些数据集上取得了很好的泛化。如果将这项研究扩展到更丰富的数据集，可以发现数据复杂性和任务难度之间更微妙的关系

附录

Breast Cancer Wisconsin (Diagnostic) Data Set:

RF:

```
clear all;
% Load the dataset
load wdbc.mat
f_train=wdbc(:,3:end);
l_train=wdbc(:,2);
%train model:RF
trees = 100; % num of trees
leaf = 5; % min leaves
OOBPrediction = 'on'; % open out of bag
OOBPredictorImportance = 'on'; % importance of features
Method = 'classification'; % task kind
net = TreeBagger(trees, f_train, l_train, 'OOBPredictorImportance',
OOBPredictorImportance,...
'Method', Method, 'OOBPrediction', OOBPrediction, 'minleaf', leaf);
view(net.Trees{1},Mode="graph");
plot(oobError(net))
```



```
xlabel("Number of Grown Trees")
ylabel("Out-of-Bag Classification Error")
oobLabels = oobPredict(net);
ind = randsample(length(oobLabels),10);
table(l_train(ind,:),oobLabels(ind,:),...
      variableNames=["TrueLabel" "PredictedLabel"])
```

Bagging:

```
clear all;
% Load the dataset
load wdbc.mat
f_train=wdbc(:,3:end);
l_train=wdbc(:,2);
%train model:Bagging
t=templateTree('MinLeafSize',5);
net= fitensemble(f_train,l_train,'Bag',100,t,'Type','classification','kFold',5);
kflc = kfoldLoss(net,'Mode','cumulative');
figure;
plot(kflc);
ylabel('10-fold Misclassification rate');
xlabel('Learning cycle');
```

Wine Dataset:

RF:

```
clear all;
% Load the dataset
load wine.mat
f_train=wine(:,2:end);
l_train=wine(:,1);
%train model:RF
trees = 100; % num of trees
leaf = 5; % min leaves
OOBPrediction = 'on'; % open out of bag
OOBPredictorImportance = 'on'; % importance of features
Method = 'classification'; % task kind
net = TreeBagger(trees, f_train, l_train, 'OOBPredictorImportance',
OOBPredictorImportance,...
'Method', Method, 'OOBPrediction', OOBPrediction, 'minleaf', leaf);
view(net.Trees{1},Mode="graph");
plot(oobError(net))
xlabel("Number of Grown Trees")
ylabel("Out-of-Bag Classification Error")
oobLabels = oobPredict(net);
ind = randsample(length(oobLabels),10);
table(l_train(ind,:),oobLabels(ind,:),...
      variableNames=["TrueLabel" "PredictedLabel"])
```

Bagging:

```

clear all;
% Load the dataset
load wine.mat
f_train=wine(:,2:end);
l_train=wine(:,1);
%train model:Bagging
t=templateTree("MinLeafSize",5);
net= fitensembles(f_train,l_train,'Bag',100,t,'Type','classification','kFold',5);
kflc = kfoldLoss(net,'Mode','cumulative');
figure;
plot(kflc);
ylabel('10-fold Misclassification rate');
xlabel('Learning cycle');

```

Student Performance Dataset:

RF:

```

clear all;
% Load the dataset
load student.mat
f_train=student(:,1:end-1);
l_train=student(:,end);
%train model:RF
trees = 100; % num of trees
leaf = 5; % min leaves
OOBPrediction = 'on'; % open out of bag
OOBPredictorImportance = 'on'; % importance of features
Method = 'regression'; % task kind
net = TreeBagger(trees, f_train, l_train, 'OOBPredictorImportance',
OOBPredictorImportance,...
'Method', Method, 'OOBPrediction', OOBPrediction, 'minleaf', leaf);
view(net.Trees{1},Mode="graph");
plot(oobError(net))
xlabel("Number of Grown Trees")
ylabel("Generalization Error")
oobLabels = oobPredict(net);
ind = randsample(length(oobLabels),10);
table(l_train(ind,:),oobLabels(ind,:),...
VariableNames=["TrueLabel" "PredictedLabel"])

```

Bagging:

```

clear all;
% Load the dataset
load student.mat
f_train=student(:,1:end-1);
l_train=student(:,end);
%train model:Bagging
t=templateTree("MinLeafSize",5);
net= fitensembles(f_train,l_train,'Bag',100,t,'Type','regression','kFold',5);
kflc = kfoldLoss(net,'Mode','cumulative');
figure;
plot(kflc);
ylabel('Generalization Error');
xlabel('Learning cycle');

```

Computer Hardware Dataset:




RF:

```
clear all;
% Load the dataset
load computer.mat
f_train=computer(:,1:end-1);
l_train=computer(:,end);
%train model:RF
trees = 100; % num of trees
leaf = 5; % min leaves
OOBPrediction = 'on'; % open out of bag
OOBPredictorImportance = 'on'; % importance of features
Method = 'regression'; % task kind
net = TreeBagger(trees, f_train, l_train, 'OOBPredictorImportance',
OOBPredictorImportance,...
'Method', Method, 'OOBPrediction', OOBPrediction, 'minleaf', leaf);
view(net.Trees{1},Mode="graph");
plot(oobError(net))
xlabel("Number of Grown Trees")
ylabel("Generalization Error")
oobLabels = oobPredict(net);
ind = randsample(length(oobLabels),10);
table(l_train(ind,:),oobLabels(ind,:),...
VariableNames=["TrueLabel" "PredictedLabel"])
```

Bagging:

```
clear all;
% Load the dataset
load computer.mat
f_train=computer(:,1:end-1);
l_train=computer(:,end);
%train model:Bagging
t=templateTree("MinLeafSize",5);
net= fitensemble(f_train,l_train,'Bag',100,t,'Type','regression','KFold',5);
kflc = kfoldLoss(net,'Mode','cumulative');
figure;
plot(kflc);
ylabel('Generalization Error');
xlabel('Learning cycle');
```

参考文献

1. BeWolberg,William, Mangasarian,Olvi, Street,Nick, and Street,W.. (1995). Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository. <https://doi.org/10.24432/C5DW2B>. 
2. Aeberhard,Stefan and Forina,M.. (1991). Wine. UCI Machine Learning Repository. <https://doi.org/10.24432/C5PC7J>. 
3. Cortez,Paulo. (2014). Student Performance. UCI Machine Learning Repository. <https://doi.org/10.24432/C5TG7T>. 
4. Feldmesser,Jacob. (1987). Computer Hardware. UCI Machine Learning Repository. <https://doi.org/10.24432/C5830D>. 