随机森林预测细菌形态软件使用说明

# 一、功能简介

该程序首先利用 **pfam\_scan** 对每个物种编码的全部蛋白质序列进行结构域注释；随后，将每条蛋白质的结构域标签用连字符连接并汇总，生成 **domain 丰度矩阵**。结合已观测到的表型信息，对样本进行分组，并借助 **randomForest** 包实施随机森林算法，完成微生物分组判别、候选基因结构域特征筛选与结果可视化。该流程既适用于二分类，也适用于多分类任务，并提供特征重要性排序及模型交叉验证功能。

# 二、输入文件格式

## 1.分组文件（如3750group3-1.txt）

（1）行名为样本ID；

（2）必含“Type”列（类别/分组标签，需为factor类型，本项目中展示的是细菌形状数据），可含“Group”列（用于区分训练集（group1）、验证集（group2））。机器学习分组参考格式如下：

SampleID Type Group  
Species1 T1 group1  
Species 2 T2 group1  
Species 3 T1 group2  
...

## 2.丰度矩阵（如train\_matrix\_fixed.txt）

（1）行名为feature/domain（如细菌或基因）；

（2）列名为样本ID；

（3）单元格为丰度值。表型数据参考格式如下：

Species 1 Species 2 Species 3 ...  
feature1 2 3 0  
feature2 5 1 1  
...

# 三、主要步骤与命令

## 1.环境准备

#加载必要的R 包 首先判断是否存在randomForest, ggplot2, pheatmap三个软件包，如果不存在需要下载安装。

# 检查并安装缺失的包

pkgs\_needed <- c("randomForest", "ggplot2", "pheatmap")

to\_install <- pkgs\_needed[!pkgs\_needed %in% installed.packages()[, "Package"]]

if (length(to\_install)) {

install.packages(to\_install, dependencies = TRUE)

}

# 加载包

lapply(pkgs\_needed, library, character.only = TRUE)

## 2.读取与整理数据

# 读取分组和丰度矩阵

# 请先手工解压train\_matrix\_fixed.zip 文件过大github不给上传，我们压缩后上传的文件  
design = read.table("3750group3-1.txt", header = T, row.names = 1)  
design$Type = as.factor(design$Type)  
otu\_table = read.table("train\_matrix\_fixed.txt", header = T, row.names = 1)  
otu\_table = t(otu\_table) # 保证行为样本，列为feature

## 3.选择训练集

design\_sub = subset(design, Group %in% c("group1"))  
# 保证样本ID一致  
idx = rownames(design\_sub) %in% colnames(otu\_table)  
design\_sub = design\_sub[idx,]  
otu\_sub = otu\_table[, rownames(design\_sub)]

## 4.随机森林模型训练

# set.seed(3)每次运行可以手动的调整括号里面的值，多次的随机调整就可以作为手动的随机种子。

set.seed(3)  
rf = randomForest(t(otu\_sub), design\_sub$Type, importance=TRUE, proximity=TRUE, ntree=1000)  
save(rf, file = 'bestmodel\_group3\_1.RData')

## 5.特征筛选与交叉验证

set.seed(827)  
result = rfcv(t(otu\_sub), design\_sub$Type, cv.fold=5)  
save(result, file = 'best\_rfcv\_group3\_1.RData')  
error\_data <- as.data.frame(result$error.cv)  
write.table(error\_data, file = 'best\_error\_group3\_1.txt', sep = '\t', row.names = T, quote = F, col.names = NA)  
with(result, plot(n.var, error.cv, log="x", type="o", lwd=1))

## 6.训练集预测及结果导出

train.p = predict(rf, type = "response")  
df = data.frame(observed = design\_sub$Type, predict = train.p)  
write.table(df, file = "train\_predict\_group3\_1.txt", quote = F, sep = '\t', row.names = T, col.names = T)

## 7.特征重要性分析与可视化

imp = as.data.frame(rf$importance)  
imp = imp[order(imp[,1], decreasing = T), ]  
write.table(imp, file = "best\_importance\_class\_group3\_1.txt", quote = F, sep = '\t', row.names = T, col.names = T)  
  
# 可视化（柱状图，Top 10）  
varImpPlot(rf, main = "Top 10 - Feature importance", n.var = 10)  
  
# ggplot2美化  
imp = read.table("best\_importance\_class\_group3\_1.txt", header=T, row.names= 1, sep="\t")  
imp = head(imp, n=23)  
imp = imp[order(imp[,3]), ]  
imp$Domain = factor(rownames(imp), levels=rownames(imp))  
p = ggplot(data = imp, mapping = aes(x=Domain, y=MeanDecreaseAccuracy, fill=Domain)) +   
 geom\_bar(stat="identity") + coord\_flip() + theme\_bw()  
ggsave(p, filename = "imp\_shape.pdf", width = 16, height = 9)

## 8.验证集预测与评估

design\_test = subset(design, Group %in% c("group2"))  
idx = rownames(design\_test) %in% colnames(otu\_table)  
design\_test = design\_test[idx, ]  
otu\_sub = otu\_table[, rownames(design\_test)]  
otutab\_t = as.data.frame(t(otu\_sub))  
otutab\_t$Type = design[rownames(otutab\_t), ]$Type  
  
otutab.pred = predict(rf, t(otu\_sub))  
pre\_tab = table(observed=otutab\_t[,"Type"], predicted=otutab.pred)  
print(pre\_tab)  
predict = data.frame(Type = otutab\_t[,"Type"], predicted=otutab.pred)  
write.table("SampleID\t", file="RF\_prediction\_binary.txt", append=F, quote=F, eol="", row.names=F, col.names=F)  
write.table(predict, file = "RF\_prediction\_binary.txt", append=T, quote = F, row.names = T, col.names = F, sep = "\t")

# 四、常见问题与注意事项

1.样本ID须严格一致：分组表和丰度矩阵的样本ID需匹配，否则数据无法正确关联。  
2.分组变量需为因子（factor）类型：否则模型训练或评估时可能报错。  
3.输出文件覆盖：请注意输出文件的覆盖和命名，避免误操作。  
4.R环境：建议R≥4.0.0，需提前安装好所需R包（randomForest, ggplot2, pheatmap）。

# 五、结果解读

1.train\_predict\_group3\_1.txt：训练集预测与真实结果对比  
2.best\_importance\_class\_group3\_1.txt：特征贡献度排序  
3.imp\_shape.pdf：特征重要性美化图  
4.RF\_prediction\_binary.txt：验证集预测结果  
5.best\_error\_group3\_1.txt：特征数-错误率关系表

# 六、随机森林模型训练中涉及的超参数

rf = randomForest(t(otu\_sub), design\_sub$Type, importance=TRUE, proximity=TRUE, ntree = 1000)

1.ntree = 1000  
指定随机森林中生成的树的数量为1000棵。  
随机森林的原理是：每棵树都有一定“随机性”，如果树太少，模型的结果容易波动、稳定性差。增加树的数量，能够让模型结果更平滑，减少偶然性，提高泛化能力。但是会增加计算量，拖慢分析流程。在实际应用中，通常500-2000棵树之间都能保证模型稳定，1000属于中间值，既能保证重要性指标比较稳定，也不会拖慢分析流程，是较为稳妥的选择。  
  
2.importance = TRUE  
开启特征重要性评估。该参数不影响模型本身，但能输出每个特征对模型分类的贡献度（如MeanDecreaseAccuracy）。  
  
3.proximity = TRUE  
计算样本间的邻近度矩阵，可用于后续可视化或样本聚类分析。此参数同样不影响模型预测性能。  
  
4.其他超参数（如mtry、nodesize等）  
采用了默认值，即mtry为每次分裂时选择的特征数（默认为总特征数的开方），nodesize为每棵树的最小叶子节点样本数（分类任务默认为1）。采用默认值便于流程标准化，有助于实验流程统一，结果便于不同批次、不同人员之间横向对比。避免繁琐的调参过程，尤其在探索性分析或数据量较小时，提高效率。

# 七、模型验证类型

采用了交叉验证方法来进行特征筛选与验证：

result = rfcv(t(otu\_sub), design\_sub$Type, cv.fold=5)

1.使用了5折交叉验证（cv.fold=5）；  
2.训练集会被随机分成5个子集，每次用4个子集训练，1个子集验证，重复5次，得到不同特征数下的交叉验证错误率（error.cv）；  
3.通过交叉验证结果选择最佳特征数量（错误率最低处为最优），从而避免过拟合。