############################################

## Scoble, L (2023) R script for ##

## pre-processing spectral data, data analysis ##

## and classification. ##

############################################

**Table of Contents**

Set up: 2

Baseline, EMSC correction and 2nd Derivative: 4

Plot non-differentiated spectra following parts of Jardine (2021) R script: 7

PCA Analysis: 11

HCA Plot: 14

Decision Trees: 16

randomForest: 23

MDA wavenumber boxplot: 31

References: 34

###########################################################

## Set up working directory and load libraries ##

## Scoble, L and Fyfe, R, (2023) ##

###########################################################

setwd("D:\\ALL DATA2\\baseline work")

library(corrplot)

library(caret)

library(tidyverse)

library(class)

library(prospectr)

library(ggplot2)

library(grid)

library(baseline)

library(EMSC)

library(vegan)

library(dendextend)

library(circlize)

library(ape)

library(RColorBrewer)

library(randomForest)

library(dplyr)

library(tree)

library(caTools)

library(tidyverse)

library(readr)

library(rpart)

library(rpart.plot)

library(reshape2)

## stick all scans together

## requires individual files with a .dtp extension

#list all files (with .dpt extension)

file.list <- list.files(pattern = "\\.dpt$") #only lists dpt files

#make empty dataframe

df <- read.csv(file.list[1], header = F)

df <- rbind(c("wavelength","Agrostis"), df)

#loop across all files and stick together into single file

count = 1

for(i in file.list){

print(paste("count =", count, i)) #flag for progress

#read in individual file

dat <- read.csv(i, header = F)

#extract sample code from filename

sample <- gsub(".dpt", "", i)

#append the data to the dataframe

dat <- rbind(c("wavelength", sample), dat)

df <- cbind(df, dat[,2])

count = count + 1

}

#prepare the combined file for export

df <- df[,-2] #drops col 2 (duplicate data)

colnames(df) <- df[1,] #define column names as sample names

df <- df[-1,] #drop top row (the un-needed names)

#export to csv format

write.csv(df, "all.data.scans.final.csv", row.names = F)

########################################################

## Baseline, EMSC correction and 2nd Derivative ##

## Scoble, L (2023) ##

########################################################

##### Read data in #####

Species\_data <- read.csv("all.data.scans.final.csv", check.names = F)

Species\_data <- data.frame(t(Species\_data))

colnames(Species\_data) <- Species\_data[1,]

Species\_data <- Species\_data[-1,]

##### Non-differentiated spectra #####

# Baseline correction

species.baseline <- baseline(as.matrix(Species\_data), method = "modpolyfit", deg = 2)

species.corrected <- data.frame(species.baseline@corrected)

colnames(species.corrected) <- colnames(Species\_data)

species.corrected <- as.data.frame(species.corrected,

row.names = rownames(Species\_data))

# EMSC correction of baseline corrected data

Species.emsc1 <- EMSC(species.corrected, degree = 3,

reference = colMeans(species.corrected))

emsc.corrected1 <- data.frame(t(Species.emsc1$corrected))

# Write file

write.csv(emsc.corrected1, "all.data.emsc.baseline.final.csv", row.names = T)

##### Second derivative of original data #####

Species.derivtwo <- savitzkyGolay(Species\_data, p = 2, w = 15, m = 2)

Species.derivtwo <- as.data.frame(Species.derivtwo, row.names = rownames(Species\_data))

# EMSC correction of second derivative data

Species.deriv.emsc <- EMSC(Species.derivtwo, degree = 1,

reference = colMeans(Species.derivtwo))

Species.deriv.emsc <- Species.deriv.emsc$corrected

# Write file

Species.deriv.emsc.pca <- data.frame(Species.deriv.emsc)

names(Species.deriv.emsc)<-sapply(str\_remove\_all(colnames(Species.deriv.emsc),"X"),"[")

write.csv(Species.deriv.emsc.pca, "Species.deriv.emsc.final.csv")

##### Prepare derivative file for OriginLabs #####

# Remove row names into column

Species.deriv.emsco <- cbind(rownames(Species.deriv.emsc.pca), data.frame(Species.deriv.emsc.pca, row.names = NULL))

# Create new short names

Species <- c(rep("Agros", 51),

rep("Antho", 51),

rep("Desch", 51),

rep("Festu", 50),

rep("Molin", 50))

#Factorise

Species <- as.factor(Species)

#Bind the two together

Species.deriv.emsco <- cbind(Species, Species.deriv.emsco)

#Remove the sample labels

Species.deriv.emsco <- Species.deriv.emsco[,-2]

names(Species.deriv.emsco)<-sapply(str\_remove\_all(colnames(Species.deriv.emsco),"X"),"[")

# Average of each species

Species.derivo.means <- aggregate(Species.deriv.emsco[,2:1749],

by = list(Species),

FUN = mean)

rownames(Species.derivo.means) <- Species.derivo.means[,1]

Species.derivo.means <- Species.derivo.means[,-1]

# Write file

Species.derivo.means <- data.frame(t(Species.derivo.means))

write.csv(Species.derivo.means, "Origin.means.emsc.all.final.csv")

#########################################

## Plot non-differentiated spectra ##

## following parts of Jardine (2021) ##

## R script. ##

## Scoble, L (2023) ##

#########################################

# Read in file

Ad1 <- read.csv("all.data.emsc.baseline.final.csv", check.names = F, row.names = 1)

Ad1 <- data.frame(t(Ad1))

names(Ad1)<-sapply(str\_remove\_all(colnames(Ad1),"X"),"[")

Ad2 <- Ad1

Ad1 <- Ad1[1:253,]

str(Ad1)

# Remove row names into column

Ad1 <- cbind(rownames(Ad1), data.frame(Ad1, row.names = NULL))

# Create new short names

Species <- c(rep("Agros", 51),

rep("Antho", 51),

rep("Desch", 51),

rep("Festu", 50),

rep("Molin", 50))

#Factorise

Species <- as.factor(Species)

#Bind the two together

Ad1 <- cbind(Species, Ad1)

#Remove the sample labels

Ad1 <- Ad1[,-2]

##### Mean and Standard Deviation #####

grass.means <- aggregate(Ad2,

by = list(Species),

FUN = mean)

rownames(grass.means) <- grass.means[,1]

grass.means <- grass.means[,-1]

grass.sd <- aggregate(Ad2,

by = list(Species),

FUN = sd)

rownames(grass.sd) <- grass.sd[,1]

grass.sd <- grass.sd[,-1]

#### For Origin Plots #####

grass.means <- data.frame(t(grass.means))

write.csv(grass.means, "Grass.means.origin.csv")

##### Plot Data #####

grass.means <- data.frame(t(grass.means))

# Full spectra (first plot only)

par(mfrow = c(1,2), mar = c(3,2,1,0) + 0.01)

#fingerprint region (second plot only)

par(mar = c(3, 1, 1, 3) + 0.01)

col <- brewer.pal(5, "Dark2")

# Select colours from RColorBrewer

speciescol <- c("#1B9E77","#D95F02", "#7570B3", "#E7298A", "#66A61E")

yvals <- seq(from = 4.5, to = 0.05, length.out = 5)

wavenumber <- (gsub("X","",colnames(Ad1[,2:1763])))

wavenumber <- as.numeric(wavenumber)

length(wavenumber)

length(grass.means[1,])

# Change xlim value for second plot (1800-600)

plot(wavenumber, grass.means[1,], las = 1,

type = "n", xlim = c(1800, 600), ylim = c(0, 6),

xlab = "", ylab = "",

yaxt = "n", xaxt = "n")

for(i in 5:1) {

col.e <- col2rgb(speciescol[i])

polygon(c(wavenumber, rev(wavenumber)),

c(grass.means[i,]+yvals[i]+grass.sd[i,],

rev(grass.means[i,]+yvals[i]-grass.sd[i,])),

col = rgb(col.e[1], col.e[2], col.e[3], alpha = 80, maxColorValue = 255),

border = NA)

for(i in 5:1) {

lines(wavenumber, grass.means[i,]+yvals[i],

col = speciescol[i])

}}

axis(1, lwd = 0, lwd.ticks = 2, tcl = 0.3,

mgp = c(1.5, 0.2, 0),

las = 1)

mtext(expression("Wavenumber cm"^-1), side = 1, line = 1)

# For first plot only

mtext("Relative Intensity", side = 2, line = 0.6, las = 0)

# For second plot only

legend.names <- cbind.data.frame(row.names(grass.means), grass.means)

#make and populate a new column with a short species name

legend.names$spec <- substr(legend.names$`row.names(grass.means)`, 1,6)

legend.names$spec <- as.character(legend.names$spec)

leg.txt <- unique(legend.names$spec)

legend("right", inset = c(-0.25, 0), leg.txt, pch = 19, cex = 0.65,

col = col, xpd = TRUE, bty = "n")

##### ############################

## Fyfe, R and Scoble, L (2023) ##

## PCA Analysis ##

#################################

# Non differentiated spectra

dfbaseemsc <- read.csv("all.data.emsc.baseline.final.csv", check.names = F, row.names = 1)

dfbaseemsc <- data.frame(t(dfbaseemsc))

names(dfbaseemsc)<-sapply(str\_remove\_all(colnames(dfbaseemsc),"X"),"[")

# Truncate

dfbaseemsc1 <- dfbaseemsc[1141:ncol(dfbaseemsc)]

# Remove anomalies

dfbaseemsc2 <- dfbaseemsc1[-c(17, 18, 19, 90, 91, 92, 93, 95, 96, 97, 98, 99),]

#Rename to make easier for plot

df.trunc <- dfbaseemsc2

##### Second Derivative Spectra (all data) #####

df.trunc1 <- read.csv("Species.deriv.emsc.final.csv", check.names = F, row.names = 1)

# Truncate (134 instead of 1141 so data starts at same wavenumber)

df.trunc2 <- df.trunc1[1134:ncol(df.trunc1)]

df.trunc2 <- df.trunc2[-c(17, 18, 19, 90, 91, 92, 93, 95, 96, 97, 98, 99),]

names(df.trunc2)<-sapply(str\_remove\_all(colnames(df.trunc2),"X"),"[")

##### Plot PCA, replace df.trunc with df.trunc2 for second plot #####

dfsmoo.pca <- prcomp(df.trunc)

dfsmoo.pca.scores <- as.data.frame(dfsmoo.pca$x)

dfsmoo.pca.scores <- cbind.data.frame(row.names(df.trunc), dfsmoo.pca.scores[,1:5])

summary(dfsmoo.pca)

#make and populate a new column with a short species name

dfsmoo.pca.scores$spec <- substr(dfsmoo.pca.scores$`row.names(df.trunc)`, 1,6)

#make a colour code for each species using short species names

groups <- cbind.data.frame(unique(dfsmoo.pca.scores$spec),

seq(1, length(unique(dfsmoo.pca.scores$spec)), by = 1))

colnames(groups) <- c("spec", "group")

#join the colour codes to the PCA result file

dfsmoo.pca.scores <- merge(dfsmoo.pca.scores, groups, by = "spec")

col <- brewer.pal(5, "Dark2")

dfsmoo.pca.scores$group <- as.factor(dfsmoo.pca.scores$group)

par(xpd = FALSE, mfrow = c(1,1), mar = c(5, 5, 5, 7), cex = 0.5, adj = 0.5, tck = 0.01)

plot(dfsmoo.pca.scores$PC1, dfsmoo.pca.scores$PC2, group = dfsmoo.pca.scores$groups,

col = c("#1B9E77","#D95F02", "#7570B3", "#E7298A", "#66A61E")[as.factor(dfsmoo.pca.scores$group)],

pch = 19, cex = 1.5, asp = 1, cex.axis = 1.5, xlab = "PC1 (74%)", ylab = "PC2 (18%)", cex.lab = 1.5)

abline(h = 0, col = "grey")

abline(v = 0, col = "grey")

# Second derivative plot only

dfsmoo.pca.scores$spec <- as.character(dfsmoo.pca.scores$spec)

leg.txt <- unique(dfsmoo.pca.scores$spec)

legend("right", inset = c(-0.15, 0), leg.txt, pch = 19, cex = 1.5,

col = col, xpd = TRUE, bty = "n")

##### Loading Plots - Run for each PCA plot #####

loadings <- as.data.frame(dfsmoo.pca$rotation)[1:2]

scale <- min(max(abs(dfsmoo.pca.scores$PC1))/max(abs(loadings$PC1)),

max(abs(dfsmoo.pca.scores$PC2))/max(abs(loadings$PC2))) \* 0.8

#extract the wavenumbers as numbers from rotation

wavenumbers <- as.numeric(rownames(dfsmoo.pca$rotation))

#extracts the first column (PCA1). Change [,1] to [,2] for PCA2 etc.

PC1loading <- as.data.frame(loadings[,1])

PC2loading <- as.data.frame(loadings[,2])

#writes the wavenumbers to the PCA1loadings object

PC1loading$wavenumber <- wavenumbers

PC2loading$wavenumber <- wavenumbers

colnames(PC1loading) <- c("loading", "wavenumber")

colnames(PC2loading) <- c("loading", "wavenumber")

#switch PC1loadings to PC2 for other plot

plot(PC1loading$loading ~ PC1loading$wavenumber, type = "l",

xlim = c(1800,600), xlab = "Wavenumber", ylab = "PC1 Loadings", cex.axis = 1.5,

cex.lab = 1.5)

#2nd deriv line

abline(h = 0, col = "black")

# Write files for loadings

write.csv(PC1loading, "PC1Loading.csv")

write.csv(PC2loading, "PC2Loading.csv")

#####################################

## Fyfe, R and Scoble, L (2023) ##

## HCA Plot - Repeat For Each Set ##

## of Data (df.trunc/df.trunc2) ##

#####################################

diss <- dist(df.trunc2, method = "euclidean")

# Cluster analysis

cluster <- as.dendrogram(hclust(diss))

# Set plotting margins and font size for the general plots

par(cex=0.5, mar=c(5, 8, 4, 1))

# c=Choose number of clusters, 5 separates the main species

k = 5

# Set up plotting colours

cluster <- cluster %>%

color\_branches(k = k) %>%

color\_labels(k = k)

# Plot circular dendrogram

circlize\_dendrogram(cluster)

# Export the cluster numbers assigned to samples

cuts <- cbind.data.frame(rownames(df.trunc), cutree(cluster, k = k))

colnames(cuts) <- c("sample", "cluster\_number")

write.csv(cuts, "cluster.groups.by.sample.diff.final.csv", row.names = F)

# ITOL file

my\_tree <- as.phylo(cluster)

write.tree(phy = my\_tree, file = "Treefinal.diff.newick")

#####################################

## Scoble, L (2023) ##

## Decision Trees and randomForest ##

#####################################

##### PART 1 - Decision trees: Extracting rpart rules which show which wavenumbers

# are causing discrepancies between species - then compare to PCA loading plots #####

# Read in file

d <- read.table("all.data.emsc.baseline.final.csv", sep = ",", header = T, row.names = 1)

d <- data.frame(t(d))

names(d)<-sapply(str\_remove\_all(colnames(d),"X"),"[")

# Truncate spectra

d <- d[,1141:ncol(d)]

d <- d[1:253,]

str(d)

d <- cbind(rownames(d), data.frame(d, row.names = NULL))

# Make column with short specie names

Species <- c(rep("Agros", 51),

rep("Antho", 51),

rep("Desch", 51),

rep("Festu", 50),

rep("Molin", 50))

# Factorise

Species <- as.factor(Species)

# Bind the two together

d <- cbind(Species, d)

# Remove the sample labels

d <- d[,-2]

summary(d$Species)

set.seed(2)

##### First decision (classification) tree using all data #####

fit <- rpart(Species ~., data = d, method = "class")

par(mar = c(2, 4, 4, 4))

par(mfrow = c(1,1))

# Plot classification tree

plot(fit)

text(fit, cex = 0.9, xpd = TRUE)

# Use rplot for more better visuals (legend position may need to be changed)

rplot <- rpart.plot(fit, type = 4, extra = "auto", clip.right.labs = FALSE,

legend.x = 0.85, legend.y = 1, legend.cex = 1.3,

cex = 0.8)

# Extract the rules that the algorithm uses to build tree and splits

# This is to look at what wavenumbers are driving the discrepancy between

# species

# Digits = 3 to get an extra decimal place (easier to refer to the data)

rpart.rules(fit)

rules <- rpart.rules(fit, digit = 3)

# Remove columns that aren't relevant

rules <- rules[,-2]

rules <- rules[,-2]

rules <- rules[,-5]

rules <- rules[,-8]

rules <- rules[,-11]

# Change colnames (Less than, Equal to, Greater than (L/E/G), Absorbance units (Au))

colnames(rules) <- c("Species", "Wavenumber1", "L/E/G", "Au", "Wavenumber2",

"L/E/G", "Au", "wavenumber3", "L/E/G", "Au",

"wavenumber4", "L/E/G", "Au")

# Write csv

write.csv(rules, "rpart.wavenumber.rules.final.csv")

# Find wavenumbers in original dataset to cross check rules

WN1 <- d %>% dplyr::select(X1693.4306)

WN2 <- d %>% dplyr::select(X883.36129)

WN3 <- d %>% dplyr::select(X1745.50649)

WN4 <- d %>% dplyr::select(X1151.45566)

cross\_check <- cbind(Species, WN1, WN2, WN3, WN4)

colnames(cross\_check) <- gsub("X","",colnames(cross\_check[,1:5]))

write.csv(cross\_check, "Cross\_check\_wavenumbers.final.csv")

##### Looped Variance #####

##### Split the data and run decision tree 100 times in a loop #####

# Will the same four wavenumbers still be prominent or will splitting the data

# create more variance.

set.seed(2)

tree\_lengths <- data.frame()

for(i in 1:100) {

train <- sample(nrow(d), 0.8\*nrow(d))

training\_data <- d[train,]

dim(training\_data)

summary(training\_data$Species)

testing\_data <- d[-train, ]

dim(testing\_data)

summary(testing\_data$Species)

tree\_i <- rpart(Species ~ ., data = training\_data, method = "class")

wavesum <- tree\_i$frame$var

tree\_lengths <- rbind(tree\_lengths, wavesum)

names(tree\_lengths) <- NULL

}

par(mfrow = c(1,1))

par(mar = c(2, 4, 4, 2))

rpart.plot(tree\_i, type = 4, extra = 104, clip.right.labs = FALSE, digits = 2,

round = 0, legend.x = 0.85, legend.y = 1, legend.cex = 1,

cex = 0.7)

# Pull one tree from loop to look at rules

# digits = 3 to get an extra decimal place (easier to refer to the data)

rpart.rules(tree\_i)

rules\_one <- rpart.rules(tree\_i, digit = 3)

# Remove columns that aren't relevant

rules\_one <- rules\_one[,-2]

rules\_one <- rules\_one[,-2]

rules\_one <- rules\_one[,-5]

rules\_one <- rules\_one[,-8]

rules\_one <- rules\_one[,-11]

#change colnames (Less than, Equal to, Greater than (L/E/G), Absorbance units (Au))

colnames(rules\_one) <- c("Species", "Wavenumber1", "L/E/G", "Au", "Wavenumber2",

"L/E/G", "Au", "wavenumber3", "L/E/G", "Au",

"wavenumber4", "L/E/G", "Au" )

# What does the new set of rules for split data show compared to the previous?

write.csv(rules\_one, "rpart\_wavenumbers\_rules\_loop.final.csv")

WN5 <- d %>% dplyr::select(X1693.4306)

WN6 <- d %>% dplyr::select(X883.36129)

WN7 <- d %>% dplyr::select(X1745.50649)

WN8 <- d %>% dplyr::select(X1155.31313)

cross\_check1 <- cbind(Species, WN5, WN6, WN7, WN8)

colnames(cross\_check1) <- gsub("X","",colnames(cross\_check1[,1:5]))

write.csv(cross\_check, "Cross\_check\_wavenumbers\_loop.csv")

# Clean up the tree\_lengths data frame to only have wave numbers present

tree\_lengths <- tree\_lengths[,-9]

tree\_lengths <- tree\_lengths[,-8]

tree\_lengths <- as.data.frame(apply(tree\_lengths, 2, function(x) {

x <- gsub("X", "", x)

}))

tree\_lengths <- as.data.frame(apply(tree\_lengths, 2, function(x) {

x <- gsub("<leaf>", "0", x)

}))

# Convert to num

tree\_lengths <- type.convert(tree\_lengths, as.is = TRUE)

tree\_lengths2 <- melt(tree\_lengths, id.vars = c("V1", "V2", "V3", "V4", "V5", "V6", "V7"))

# Place all wavenumbers into one column

tree\_lengths2 <- reshape(tree\_lengths, direction = "long", sep = "", varying = 1:7)

# Remove time column

tree\_lengths2 <- tree\_lengths2[,-1]

table <- table(tree\_lengths2$V)

table <- as.data.frame(table)

# Remove zero (first row) as not relevant

table <- table[-1,]

# Arrange table so Freq is descending from largest to smallest

table2 <- table %>%

arrange(desc(Freq))

# What is table showing and how does that compare to fit and also the PCA loadings

# Plot histogram

table2 <- table2[1:10,]

table3 <- as.data.frame(table2)

par(mfrow = c(1,1))

par(mar = c(2, 4, 4, 4))

ggplot(table3, aes(x = reorder(Var1, -Freq), y = Freq, fill = rules)) +

geom\_histogram(stat = "Identity", colour = "darkblue", fill = "lightblue") +

labs(x = "Wavenumber", y = "Frequency of Appearence") +

theme(panel.grid = element\_blank(), strip.text.y = element\_blank(),

axis.text.x = element\_text(angle = 50, vjust = 1, hjust = 1, size = 11, face = "bold",

colour = "black"), axis.title.x = element\_text(size = 15), axis.title.y = element\_text(size = 14),

axis.text.y = element\_text(size = 11, face = "bold", colour = "black"),

panel.background = element\_blank())

# Write csv for table

write.csv(table3, "Final.table.loop.freq.csv")

###### Repeat for first wavenumber split #####

set.seed(2)

tree\_lengths <- data.frame()

for(i in 1:100) {

train <- sample(nrow(d), 0.8\*nrow(d))

training\_data <- d[train,]

dim(training\_data)

summary(training\_data$Species)

testing\_data <- d[-train, ]

dim(testing\_data)

summary(testing\_data$Species)

tree\_i <- rpart(Species ~ ., data = training\_data, method = "class")

wavesum <- tree\_i$frame$var[1]

tree\_lengths <- rbind(tree\_lengths, wavesum)

names(tree\_lengths) <- NULL

}

tree\_lengths <- as.data.frame(apply(tree\_lengths, 1, function(x) {

x <- gsub("X", "", x)

}))

colnames(tree\_lengths) <- "wavenumber"

tree\_lengths2 <- tree\_lengths %>% group\_by(tree\_lengths$wavenumber) %>%

count(sort = TRUE)

tree\_lengths2 <- tree\_lengths2[1:6,]

ggplot(tree\_lengths2, aes(x = reorder(`tree\_lengths$wavenumber`, -n), y = n, fill = rules)) +

geom\_histogram(stat = "Identity", colour = "darkblue", fill = "lightblue") +

labs(x = "Wavenumber", y = "Frequency of Appearence") +

theme(panel.grid = element\_blank(), strip.text.y = element\_blank(),

axis.text.x = element\_text(angle = 50, vjust = 1, hjust = 1, size = 11, face = "bold",

colour = "black"), axis.title.x = element\_text(size = 15), axis.title.y = element\_text(size = 14),

axis.text.y = element\_text(size = 11, face = "bold", colour = "black"),

panel.background = element\_blank())

# Write csv for table

write.csv(tree\_lengths2,"first.wavenumber.rule.split.csv")

##### Part 2 - RandomForest #####################

## Classification using RandomForest ##

## Build model using randomForest and training data ##

#############################################

# Bagged trees

set.seed(2)

train <- sample(nrow(d), 0.8\*nrow(d))

training\_data <- d[train,]

dim(training\_data)

summary(training\_data$Species)

testing\_data <- d[-train, ]

dim(testing\_data)

summary(testing\_data$Species)

set.seed(2)

bag.RF <- randomForest(Species ~ ., data = training\_data, mtry = 622, ntree = 100,

importance = TRUE, proximity = TRUE, do.trace = TRUE)

bag.RF

#Look at error matrix

plot(bag.RF)

print(bag.RF)

#Predict to see if trained forest will accurately predict test data

bag.tree <- predict(bag.RF, testing\_data, type = "class")

tab <- table(bag.tree, testing\_data$Species)

tab

write.csv(tab, "prediction.RF.final.csv")

(tab[1,5] + tab[5,1] / sum(tab))

#Plot the Variable importance

par(mfrow = c(1,1), mar = c(2,2,1,2))

varImpPlot(bag.RF,

n.var = 24,

type = 1,

sort = TRUE,

main = "Variable Importance Plot")

##### Looped randomForest for MDA investigations #####

set.seed(2)

# Make an empty list of 10

ls <- list()

n = 10

datalist = list()

# Pre-allocate for slightly more efficiency

datalist = vector("list", length = n)

# Run loop

for(i in 1:10) {

importance.tree <- randomForest(Species ~ ., d, ntree = 150, mtry = 24, importance = TRUE)

plot(importance.tree)

wavesum <- importance.tree$importance[,6, drop = FALSE]

datalist[[i]] <- cbind(rownames(wavesum), data.frame(wavesum, row.names = NULL))

colnames(datalist[[i]]) <- c("Wavenumber", "MeanDecreaseAccuracy")

for (i in 1:length(datalist)) {

assign(paste0("datalist", i), as.data.frame(datalist[[i]]))}

}

#repeat for each datalist

datalist1 <- datalist1 %>%

arrange(desc(MeanDecreaseAccuracy))

# Cbind all dataframes together

dataframeall <- cbind.data.frame(datalist1, datalist2, datalist3, datalist4,

datalist5, datalist6, datalist7, datalist8,

datalist9, datalist10)

# Convert to numeric

dataframeall <- type.convert(dataframeall, as.is = TRUE)

# Trim rows to only have the top 24 (24 is the square root of 622)

dataframeall <- dataframeall[1:24,]

# Rename column names

colnames(dataframeall) <- c("V1", "V2", "V3", "V4", "V5", "V6", "V7", "V8", "V9", "V10",

"V11", "V12", "V13", "V14", "V15", "V16", "V17", "V18", "V19", "V20")

# Split into Wavenumber and MDA

dataframewavenumber <- data.frame(dataframeall[, c(1, 3, 5, 7, 9, 11, 13, 15, 17, 19)])

dataframeMDA <- data.frame(dataframeall[, c(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)])

# Rename column names

colnames(dataframewavenumber) <- c("V1", "V2", "V3", "V4", "V5", "V6", "V7", "V8", "V9", "V10")

# Place all wavenumbers into one column

dataframewavenumber1 <- melt(dataframewavenumber, id.vars = c("V1", "V2", "V3", "V4", "V5",

"V6", "V7", "V8", "V9", "V10"))

dataframewavenumber1 <- reshape(dataframewavenumber1 , direction = "long",

sep = "", varying = 1:10)

# Rename column names

colnames(dataframeMDA) <- c("V1", "V2", "V3", "V4", "V5", "V6", "V7", "V8",

"V9", "V10")

# Place all MDA into one column

dataframeMDA1 <- melt(dataframeMDA, id.vars = c("V1", "V2", "V3", "V4",

"V5","V6", "V7", "V8", "V9", "V10"))

dataframewaveMDA1 <- reshape(dataframeMDA1 , direction = "long",

sep = "", varying = 1:10)

# Combine the Wavenumber and MDA column from each dataframe

combinedWNMDA <- cbind.data.frame(dataframewavenumber1$V, dataframewaveMDA1$V)

# Rename

colnames(combinedWNMDA) <- c("Wavenumber", "MDA")

# Remove "X" character

combinedWNMDA <- as.data.frame(apply(combinedWNMDA, 2, function(x) {

x <- gsub("X", "", x) }))

# Convert to numeric

combinedWNMDA <- type.convert(combinedWNMDA, as.is = TRUE)

# Arrange in descending order of MDA numbers

combinedWNMDA <- combinedWNMDA %>%

arrange(desc(MDA))

# Select only top 24 of all dataframes combined

table <- combinedWNMDA[1:24,]

table2 <- as.data.frame(table)

table2 <- table2 %>% arrange(MDA)

par(mar = c(3, 1 , 0 ,1))

# Plot dotcharts

dotchart(table2$MDA, table2$Wavenumber, xlim = range(table2$MDA),

xlab = "MeanDecreaseAccuracy", mgp=c(2,1,.5), las=1, cex = 0.9)

#### Run Rf using isolated variables #####

table2$Wavenumber

set.seed(2)

isolated <- d %>% dplyr::select(Species, X1691.50187, X1676.07197,

X1641.35472, X1467.76844, X1461.98223, X1450.40981,

X1134.09703, X1072.37747, X1068.51999, X866.00267, X858.28772,

X821.64173, X819.71299, X815.85552, X800.42563, X794.63942,

X786.92447, X727.13364, X688.55891, X632.62556, X626.83935,

X622.98187, X613.33819, X607.55198)

train <- sample(nrow(isolated), 0.8\*nrow(isolated))

training\_data1 <- isolated[train,]

dim(training\_data1)

summary(training\_data1$Species)

testing\_data1 <- isolated[-train, ]

dim(testing\_data1)

summary(testing\_data1$Species)

set.seed(2)

isolated.rf <- randomForest(Species ~ ., data = training\_data1, ntree = 100,

importance = TRUE, proximity = TRUE, do.trace = TRUE)

isolated.rf

#Look at error matrix

plot(isolated.rf)

print(isolated.rf)

#Predict test data

bag.tree <- predict(isolated.rf, testing\_data, type = "class")

tab <- table(bag.tree, testing\_data$Species)

tab

write.csv(tab, "prediction.RF.isolated.csv")

(tab[1,5] + tab[5,1] / sum(tab))

#Plot the Variable importance

par(mar = c(3, 1 , 1 ,1))

varImpPlot(isolated.rf,

type = 1,

sort = TRUE,

main = "Variable Importance Plot",

cex = 0.75)

# Run with all isolated data

set.seed(2)

isolated.rf1 <- randomForest(Species ~ ., data = isolated, ntree = 100,

importance = TRUE, proximity = TRUE, do.trace = TRUE)

isolated.rf1

#Look at error matrix

plot(isolated.rf1)

print(isolated.rf1)

#Plot the Variable importance

par(mar = c(4, 1 , 1 ,1))

varImpPlot(isolated.rf1,

type = 1,

sort = TRUE,

main = "Variable Importance Plot")

par(mar = c(3, 3, 1, 3))

# Plot final dotchart

# All data

varImpPlot(isolated.rf1,

type = 1,

sort = TRUE,

main = "Variable Importance Plot",

cex = 0.75,

mgp=c(2,1,.5))

#See if trained data can predict unlabelled test data

#make copy of testing\_data

testing\_data1 <- testing\_data

# Actual Species names

Species\_1 <- testing\_data1[1]

#Remove the sample labels

testing\_data1 <- testing\_data1[,-1]

#Unlabel data

new\_data <- data.frame(testing\_data1[,-1])

#Predict for accuracy

new\_data$predictedlabel <- predict(isolated.rf, new\_data)

new\_data$predictedlabel

Predicted <- as.character(new\_data$predictedlabel)

Actual <- as.character(testing\_data1$Species)

#Cbind the predicted labels with the known species labels from test data

new\_data1 <- as.data.frame(cbind(Predicted, Species\_1))

#View results as csv

write.csv(new\_data1, "prediciton\_name\_data\_isolated\_final.csv")

#############################################

## Plot MDA wavenumbers with relative intensity ##

## as a boxplot ##

## Scoble, L (2023) ##

############################################

impdf <- data.frame(importance(isolated.rf1))

#Remove X from dataframe

rownames(impdf) <- (gsub("X","",rownames(impdf[1:7])))

impdf <- cbind(rownames(impdf), data.frame(impdf, row.names = NULL))

# Convert to numeric

impdf <- type.convert(impdf, as.is = TRUE)

impdf <- impdf[,-8]

# Arrange data in desc of MDA

impdf <- impdf %>%

arrange(desc(MeanDecreaseAccuracy))

impdf <- impdf[,-7]

#Prepare dataframe for boxplot

impdf <- data.frame(t(impdf))

colnames(impdf) <- impdf[1,]

impdf <- impdf[-1,]

impdf <- cbind(rownames(impdf), data.frame(impdf, row.names = NULL))

# Create new Species labels

Species <- c(rep("Agros", 1),

rep("Antho", 1),

rep("Desch", 1),

rep("Festu", 1),

rep("Molin", 1))

# Factorise

Species <- as.factor(Species)

# Bind the two together

impdf <- cbind(Species, impdf)

# Remove the sample labels

impdf <- impdf[,-2]

colnames(impdf) <- (gsub("X","",colnames(impdf)))

#Melt all data together

melt <- melt(impdf)

#Plot boxplot of MDA as x axis

p <- ggplot(melt, aes(factor(variable), value, fill = Species))

p + geom\_boxplot() + facet\_wrap(~variable, scale="free") +

theme(axis.text.x = element\_blank())

#boxplot of wavenumbers in order of MDA dotchart

colnames(impdf)

varimporder <- d %>% dplyr::select(Species, X1641.35472, X786.92447, X622.98187,

X1676.07197, X727.13364, X1450.40981,

X1072.37747, X1691.50187, X1467.76844,

X1134.09703, X794.63942, X688.55891, X1461.98223,

X800.42563, X866.00267, X821.64173, X819.71299,

X626.83935, X613.33819, X815.85552, X632.62556,

X858.28772, X1068.51999, X607.55198)

colnames(varimporder) <- (gsub("X","",colnames(varimporder)))

#melt all the data together

melt <- melt(varimporder)

boxplot(melt, value ~ variable)

p <- ggplot(melt, aes(factor(variable), value, fill = Species))

p + geom\_boxplot() + facet\_wrap(~variable, scale="free") +

labs(x = "Wavenumber", y = "Relative Intensity") +

theme(axis.text.x = element\_blank())

**References**

Jardine, E. P., 2021. *Data and code for "Sporopollenin chemistry and its durability in the geological record: an integration of extant and fossil chemical data across the seed plants".* [Online] Available at: https://doi.org/10.6084/m9.figshare.11382102.v1  
[Accessed 28 September 2023].