Monte Carlo Simulation Tools for REDD+ Uncertainty Estimates

A tentative workflow for estimating uncertainty using Monte Carlo simulation

## Environment setup

# some parent packages need github sourcing   
#install.packages(c("glue","remotes"))  
#remotes::install\_github("ropensci/allodb")   
  
# note: warning messages turned off for this setup chunk  
easypackages::packages(  
 "animation", “BIOMASS", “caret", “dataMaid", “DescTools", “dplyr",  
 "extrafont", “FawR", “ForestToolsRS", "ggplot2", "htmltools",  
 "janitor", "jsonlite", "lattice", "kableExtra", "kernlab",  
 "knitr", "Mlmetrics", "olsrr", "plotly", "psych", "RColorBrewer",  
 "rmarkdown", "readxl", "tibble", "tidymodels", "tidyverse",  
 "tinytex", "tune", "useful", "webshot", "webshot2",  
 prompt = F  
 )

knitr::opts\_chunk$set(  
 echo = TRUE,   
 message = FALSE,   
 warning = FALSE,  
 error = FALSE,   
 comment = NA,   
 tidy.opts = list(width.cutoff = 60)  
)  
  
options(htmltools.dir.version = FALSE, htmltools.preserve.raw = FALSE)  
sf::sf\_use\_s2(use\_s2 = FALSE)

## Introduction

Figure 1: Visualization of cross-validation regime similar to Monte Carlo simulation. Note differences in replacement, repeatedness and odd vs

The ART-TREES (Architecture for REDD+ Transactions – The REDD+ Environmental Excellence Standard) program sets rigorous methodological requirements for quantifying emissions reductions and removals in REDD+ projects. These standards emphasize accuracy, transparency, consistency, and completeness in carbon accounting, making the integration of advanced tools and methodologies essential. This pilot test script aligns with these principles, focusing on the application of Monte Carlo simulations, machine learning models, and database management systems to meet ART-TREES requirements effectively.

The consultant, Seamus Murphy, will provide technical expertise across three key projects in the Ecosystem Services division’s REDD+ portfolio, addressing urgent staffing needs. These activities include developing uncertainty estimation pathways consistent with ART-TREES, implementing jurisdictional-scale risk mapping, and ensuring that carbon accounting systems align with internationally accepted best practices. By leveraging Monte Carlo simulations and other state-of-the-art tools, this work will contribute to robust Monitoring, Reporting, and Verification (MRV) systems required under ART-TREES.

This script outlines a structured approach to fulfilling project deliverables while adhering to the ART-TREES methodological framework, ensuring that all activities contribute to credible and scalable REDD+ outcomes.

### Scope of Work

This report aligns with specific deliverables relating to Monte Carlo support contracted under the project **Guyana (SPEC IN FOREST CARB MON - 10026.DRCT.00000.00000)**. In addition, this tentative workflow aims to align with ART-TREES methodological requirements of transparency, rigor, and alignment with jurisdictional and international standards. In addition, this report seeks feedback on the style of documentation of tools in efforts to address the contract’s capacity building targets. Specifically, these include the following objectives:

**Monte Carlo Simulation for Uncertainty Estimation**

* Develop Monte Carlo simulation pathways to quantify uncertainty in emission factors and activity data, ensuring consistency with ART-TREES’s emphasis on robust uncertainty analysis.
* Use R or other software to create systems that streamline data workflows and enhance accessibility for MRV purposes. Monte Carlo Simulation for Uncertainty Estimation
* Document methodologies and provide results in formats compliant with ART-TREES reporting standards.

**Reporting and Training Materials**

* Prepare technical reports that detail uncertainty estimation methods and database management workflows.
* Develop and deliver training materials to strengthen stakeholder capacity to use ART-TREES-aligned tools and methodologies.

Current excel tool design published [here](https://www.artredd.org/wp-content/uploads/2021/12/MC-4-estimating-ER-from-forests-update-1-1.xlsx)

## Import data

This section outlines the tools for importing and preparing forestry and biomass data for analysis, a key step in building ART-TREES-compliant MRV systems. Using the allodb package, we load a global allometry database and a dummy dataset from the Smithsonian Institute ForestGEO project.

library("allodb") # https://docs.ropensci.org/allodb/  
set.seed(333)  
#data(ufc) # spuRs::vol.m3(dataset$dbh.cm, dataset$height.m, multiplier = 0.5)  
data(scbi\_stem1)  
dataset = scbi\_stem1  
head(dataset) |> kbl(  
 caption = "Table 1: Dataset from Smithsonian Institute provided by allodb package (n = 2287") |>  
 kable\_styling()

Table 1: Dataset from Smithsonian Institute provided by allodb package (n = 2287

| treeID | stemID | dbh | genus | species | Family |
| --- | --- | --- | --- | --- | --- |
| 2695 | 2695 | 1.41 | Acer | negundo | Sapindaceae |
| 1229 | 38557 | 1.67 | Acer | negundo | Sapindaceae |
| 1230 | 1230 | 1.42 | Acer | negundo | Sapindaceae |
| 1295 | 32303 | 1.04 | Acer | negundo | Sapindaceae |
| 1229 | 32273 | 2.47 | Acer | negundo | Sapindaceae |
| 66 | 31258 | 2.19 | Acer | negundo | Sapindaceae |

psych::describe(dataset)

# A tibble: 6 × 13  
 vars n mean sd median trimmed mad min max range skew  
 <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 1 2287 2779. 1.93e3 2.52e3 2.71e3 2.09e+3 1 6.21e3 6.21e3 0.272  
2 2 2287 16577. 1.62e4 5.02e3 1.57e4 5.75e+3 1 4.02e4 4.02e4 0.396  
3 3 2287 5.52 1.08e1 1.67e0 2.65e0 7.86e-1 1 9.20e1 9.10e1 3.81   
4 4 2287 16.4 6.52e0 1.8 e1 1.67e1 0 1 3.1 e1 3 e1 -0.571  
5 5 2287 13.3 9.60e0 8 e0 1.13e1 0 1 4 e1 3.9 e1 1.59   
6 6 2287 13.1 4.02e0 1.3 e1 1.33e1 0 1 2.2 e1 2.1 e1 -0.576  
# ℹ 2 more variables: kurtosis <dbl>, se <dbl>

str(dataset)

tibble [2,287 × 6] (S3: tbl\_df/tbl/data.frame)  
 $ treeID : int [1:2287] 2695 1229 1230 1295 1229 66 2600 4936 1229 1005 ...  
 $ stemID : int [1:2287] 2695 38557 1230 32303 32273 31258 2600 4936 36996 1005 ...  
 $ dbh : num [1:2287] 1.41 1.67 1.42 1.04 2.47 ...  
 $ genus : chr [1:2287] "Acer" "Acer" "Acer" "Acer" ...  
 $ species: chr [1:2287] "negundo" "negundo" "negundo" "negundo" ...  
 $ Family : chr [1:2287] "Sapindaceae" "Sapindaceae" "Sapindaceae" "Sapindaceae" ...

## Probability density functions

Accurate selection of probability density functions (PDFs) is essential for modeling uncertainties in carbon stocks and activity data. This section describes methodologies for fitting PDFs to data, ensuring results are robust and aligned with ART-TREES best practices.

* Use of statistical tests for goodness-of-fit validation.
* Integration of domain expertise to refine parameter selection.

# add allometry database  
data(equations)  
data("equations\_metadata")  
show\_cols = c("equation\_id", "equation\_taxa", "equation\_allometry")  
eq\_tab\_acer = new\_equations(subset\_taxa = "Acer")  
kable(head(eq\_tab\_acer[, show\_cols]))  
# Compute above ground biomass  
dataset$agb = allodb::get\_biomass(  
 dbh = dataset$dbh,  
 genus = dataset$genus,  
 species = dataset$species,  
 coords = c(-78.2, 38.9)  
 )  
  
# examine dbh ~ agb function  
dbh\_agb = lm(dbh ~ agb, data = dataset)  
#olsrr::ols\_test\_breusch\_pagan(lm(dbh\_agb)) #<0.0000  
#h = lattice::histogram(dbh ~ agb, data = dataset)  
plot(  
 x = dataset$dbh,  
 y = dataset$agb,  
 col = factor(scbi\_stem1$genus),  
 xlab = "DBH (cm)",  
 ylab = "AGB (kg)"  
)  
# examine univariate distributions  
h1 = hist(dataset$dbh, breaks=10, col="red")  
xfit<-seq(min(dataset$dbh),max(dataset$dbh),length=40)  
yfit<-dnorm(xfit,mean=mean(dataset$dbh),sd=sd(dataset$dbh))  
yfit <- yfit\*diff(h1$mids[1:2])\*length(dataset$dbh)  
lines(xfit, yfit, col="blue", lwd=2)  
h2 = hist(dataset$agb, breaks=10, col="red")  
xfit<-seq(min(dataset$agb),max(dataset$agb),length=40)  
yfit<-dnorm(xfit,mean=mean(dataset$agb),sd=sd(dataset$agb))  
yfit <- yfit\*diff(h2$mids[1:2])\*length(dataset$agb)  
lines(xfit, yfit, col="blue", lwd=2)  
wilcox.test(dataset$dbh) # p<0.00001  
wilcox.test(dataset$agb) # p<0.00001

This section introduces the design of the Monte Carlo simulation regime, including:

* Simulation parameters are defined to balance computational efficiency and statistical robustness.
* Cross-validation techniques are employed to evaluate model performance and identify bias or variance.

The LGOCV acronym used in the caret package functions below stands for “leave one group out cross validation”. We must select the % of test data that is set out from the build upon which the model will be repeatedly trained. Note, the following code applies functions to full dataset without explicit training-test split. **Questions remains on whether we require cross-validation uncertainty estimate to review internal bias, and whether we would like to develop Monte Carlo tools for spatial uncertainty used in Activity Data analysis**. For your consideration, the consultant has previously developed Monte Carlo tools for LULC applications, saved [here](https://github.com/seamusrobertmurphy/02-lulc-classification)

# Cross-validation split for bias detection  
#samples = caret::createDataPartition(dataset\_tidy$volume, p = 0.80, list = FALSE)  
#train\_data = dataset\_tidy[samples, ]  
#test\_data = dataset\_tidy[-samples, ]  
  
# Simulation pattern & regime  
monte\_carlo = trainControl(  
 method = "LGOCV",  
 number = 10, # number of simulations  
 p = 0.8) # percentage resampled  
  
  
# Training model fit with all covariates (".") & the simulation  
lm\_monte\_carlo = train(  
 data = dataset,   
 agb ~ .,   
 na.action = na.omit,  
 trControl = monte\_carlo)  
  
lm\_monte\_carlo

Random Forest   
  
2287 samples  
 6 predictor  
  
No pre-processing  
Resampling: Repeated Train/Test Splits Estimated (10 reps, 80%)   
Summary of sample sizes: 1832, 1832, 1832, 1832, 1832, 1832, ...   
Resampling results across tuning parameters:  
  
 mtry RMSE Rsquared MAE   
 2 334.91964 0.5977225 114.373822  
 47 83.37237 0.9711580 14.009351  
 93 49.98649 0.9895214 8.593528  
  
RMSE was used to select the optimal model using the smallest value.  
The final value used for the model was mtry = 93.

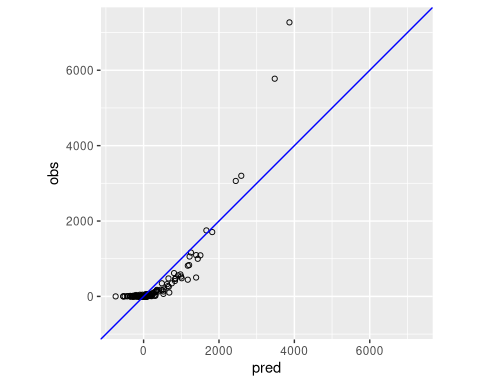
## Plot residuals

To enable access to these predictions, we need to instruct caret to retain the resampled predictions by setting savePredictions = "final" in our trainControl() function. It’s important to be aware that if you’re working with a large dataset or numerous resampling iterations, the resulting train() object may grow significantly in size. This happens because caret must store a record of every row, including both the observed values and predictions, for each resampling iteration. By visualizing the results, we can offer insights into the performance of our model on the resampled data.

monte\_carlo\_viz = trainControl(  
 method = "LGOCV",   
 p = 0.8,   
 number = 1, # just for saving previous results  
 savePredictions = "final")   
  
lm\_monte\_carlo\_viz = train(  
 agb ~ .,   
 data = dataset,   
 method = "lm",  
 na.action = na.omit,  
 trControl = monte\_carlo\_viz)  
  
head(lm\_monte\_carlo\_viz$pred) # residuals

# A tibble: 6 × 5  
 intercept pred obs rowIndex Resample   
 <lgl> <dbl> <dbl> <int> <chr>   
1 TRUE -39.3 0.282 2 Resample1  
2 TRUE -8.62 0.766 5 Resample1  
3 TRUE -31.9 0.564 6 Resample1  
4 TRUE -97.2 0.183 10 Resample1  
5 TRUE 356. 162. 20 Resample1  
6 TRUE 1394. 1095. 22 Resample1

lm\_monte\_carlo\_viz$pred |>   
 ggplot(aes(x=pred,y=obs)) +  
 geom\_point(shape=1) +   
 geom\_abline(slope=1, colour='blue') +  
 coord\_obs\_pred()



## Uncertainty Estimates

This section discusses the trade-offs and methodological choices in uncertainty estimation using Monte Carlo simulations. It aligns with ART-TREES principles by:

* Quantifying confidence intervals for emissions estimates.
* Addressing potential biases in the modeling process.
* Ensuring robustness in uncertainty reporting.

***Working Notes…***

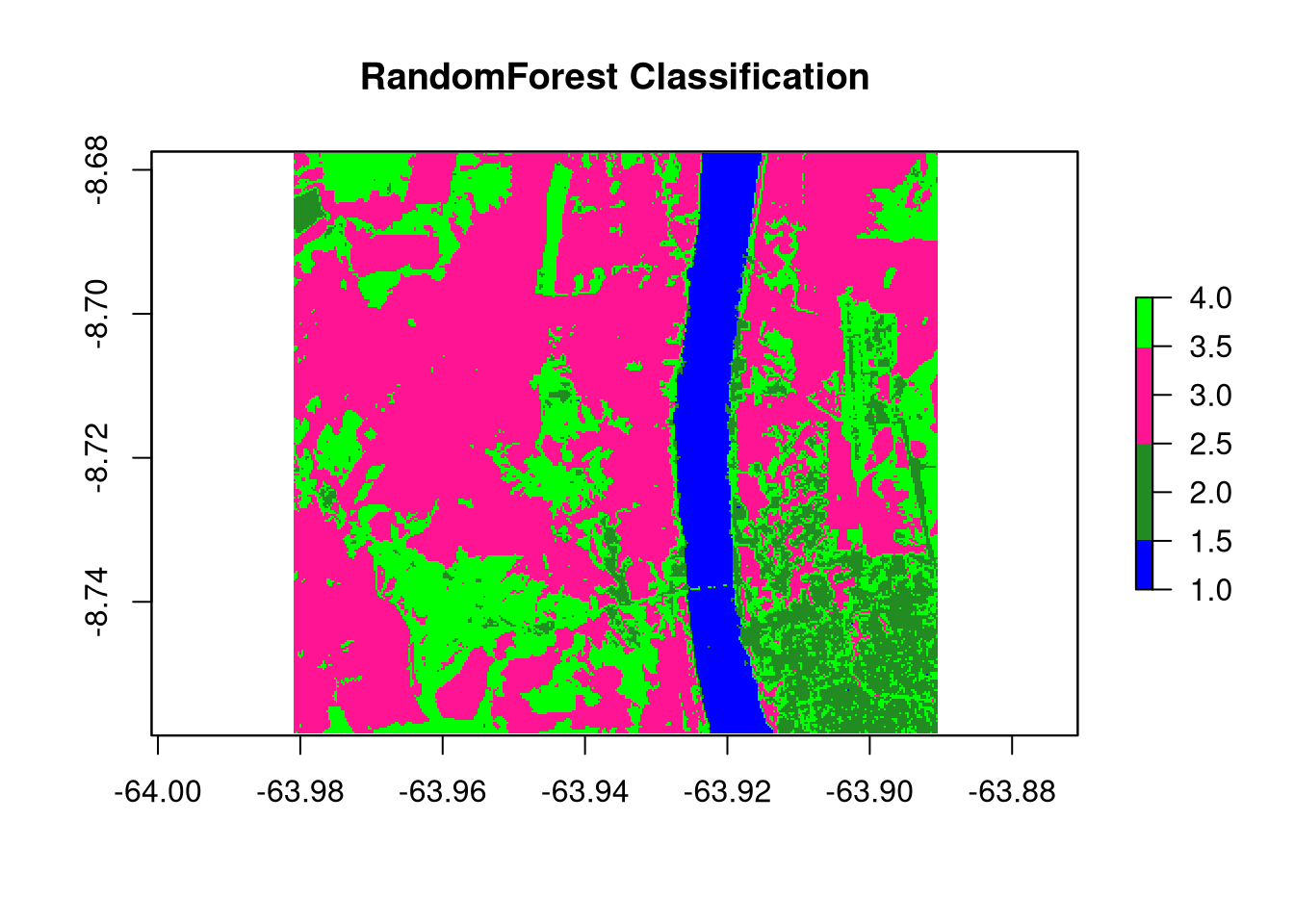
References to key studies on cross-validation methods provide a theoretical foundation for the approach.**Monte Carlo cross-validation** (MCCV) involves randomly dividing the dataset into two parts: a training subset and a validation subset, without reusing data points. The model is trained on the training subset, denoted as ( n\_t ), and assessed on the validation subset, ( n\_v ). While there are ( ) distinct ways to form the training subsets, MCCV bypasses the computational burden of evaluating all these combinations by sampling a smaller number of iterations. Zhang [3] demonstrated that performing MCCV for ( N ) iterations yields results comparable to exhaustive cross-validation over all possible subsets. However, studies investigating MCCV for large dataset sizes (( N )) remain limited.

The trade-off between bias and variance in MCCV is influenced by the choice of ( k ) (iterations) and ( n\_t ) (training subset size). Increasing ( k ) or ( n\_t ) tends to reduce bias but increases variance. Larger training subsets lead to greater similarity across iterations, which can result in overfitting to the training data. For a deeper analysis, see [2]. The bias-variance characteristics of ( k )-fold cross-validation (kFCV) and MCCV differ, but their bias levels can be aligned by selecting appropriate values for ( k ) and ( n\_t ). A detailed comparison of the bias and variance for both approaches can be found in [1], where MCCV is referred to as the “repeated-learning testing-model.”

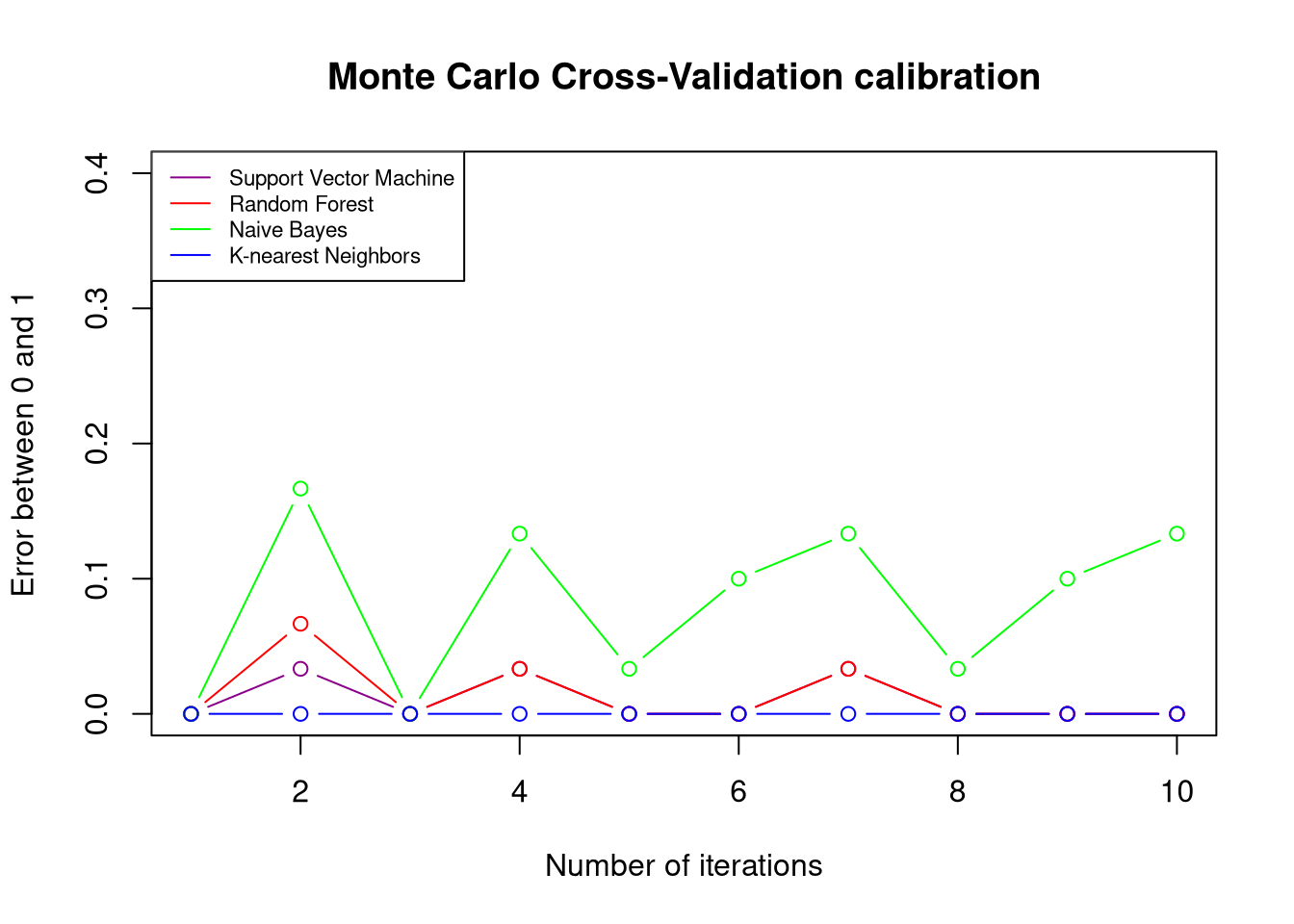
Monte Carlo Simulation in LULC Classification of Activity Data Assessment:

library(ForesToolboxRS)  
dir.create("./data/testdata")  
download.file("https://github.com/ytarazona/ft\_data/raw/main/data/LC08\_232066\_20190727\_SR.zip",destfile = "testdata/LC08\_232066\_20190727\_SR.zip")  
unzip("testdata/LC08\_232066\_20190727\_SR.zip", exdir = "testdata") download.file("https://github.com/ytarazona/ft\_data/raw/main/data/signatures.zip", destfile = "testdata/signatures.zip")  
unzip("testdata/signatures.zip", exdir = "testdata")  
  
image <- stack("./data/testdata/LC08\_232066\_20190727\_SR.tif")  
sig <- read\_sf("./data/testdata/signatures.shp")  
classRF <- mla(img = image, model = "randomForest", endm = sig, training\_split = 80)  
  
# Results  
print(classRF)

# Classification  
colmap <- c("#0000FF","#228B22","#FF1493", "#00FF00")  
plot(classRF$Classification, main = "RandomForest Classification", col = colmap, axes = TRUE)



plot(  
 cal\_ml$svm\_mccv,  
 main = "Monte Carlo Cross-Validation calibration",  
 col = "darkmagenta",  
 type = "b",  
 ylim = c(0, 0.4),  
 ylab = "Error between 0 and 1",  
 xlab = "Number of iterations"  
)  
lines(cal\_ml$randomForest\_mccv, col = "red", type = "b")  
lines(cal\_ml$naiveBayes\_mccv, col = "green", type = "b")  
lines(cal\_ml$knn\_mccv, col = "blue", type = "b")  
legend(  
 "topleft",  
 c(  
 "Support Vector Machine",  
 "Random Forest",  
 "Naive Bayes",  
 "K-nearest Neighbors"  
 ),  
 col = c("darkmagenta", "red", "green", "blue"),  
 lty = 1,  
 cex = 0.7  
)



[1] Burman, P. (1989). A comparative study of ordinary cross-validation, ( v )-fold cross-validation, and the repeated learning testing-model methods. *Biometrika*, **76**, 503–514.

[2] Hastie, T., Tibshirani, R., & Friedman, J. (2011). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer.

[3] Zhang, P. (1993). Model selection via multifold cross-validation. *Annals of Statistics*, **21**, 299–313.

devtools::session\_info()

─ Session info ───────────────────────────────────────────────────────────────  
 setting value  
 version R version 4.4.2 (2024-10-31)  
 os Fedora Linux 40 (Workstation Edition)  
 system x86\_64, linux-gnu  
 ui X11  
 language (EN)  
 collate en\_CA.UTF-8  
 ctype en\_CA.UTF-8  
 tz America/Vancouver  
 date 2024-12-19  
 pandoc 3.1.3 @ /usr/libexec/rstudio/bin/pandoc/ (via rmarkdown)  
  
─ Packages ───────────────────────────────────────────────────────────────────  
 package \* version date (UTC) lib source  
 abind 1.4-8 2024-09-12 [2] CRAN (R 4.4.1)  
 allodb \* 0.0.1.9000 2024-12-19 [1] Github (ropensci/allodb@4207f86)  
 animation \* 2.7 2021-10-07 [2] CRAN (R 4.4.0)  
 assertthat 0.2.1 2019-03-21 [2] CRAN (R 4.4.0)  
 backports 1.5.0 2024-05-23 [2] CRAN (R 4.4.0)  
 BIOMASS \* 2.1.11 2023-09-29 [2] CRAN (R 4.4.0)  
 boot 1.3-31 2024-08-28 [2] CRAN (R 4.4.1)  
 broom \* 1.0.7 2024-09-26 [2] CRAN (R 4.4.1)  
 cachem 1.1.0 2024-05-16 [2] CRAN (R 4.4.0)  
 car 3.1-3 2024-09-27 [2] CRAN (R 4.4.1)  
 carData 3.0-5 2022-01-06 [2] CRAN (R 4.4.0)  
 caret \* 7.0-1 2024-12-10 [2] CRAN (R 4.4.2)  
 cellranger 1.1.0 2016-07-27 [2] CRAN (R 4.4.0)  
 chromote 0.3.1 2024-08-30 [2] CRAN (R 4.4.1)  
 class 7.3-22 2023-05-03 [2] CRAN (R 4.4.0)  
 classInt 0.4-10 2023-09-05 [2] CRAN (R 4.4.0)  
 cli 3.6.3 2024-06-21 [2] CRAN (R 4.4.0)  
 codetools 0.2-20 2024-03-31 [2] CRAN (R 4.4.0)  
 colorspace 2.1-1 2024-07-26 [2] CRAN (R 4.4.1)  
 CoprManager 0.5.7 2024-10-31 [4] local  
 data.table 1.16.4 2024-12-06 [2] CRAN (R 4.4.2)  
 dataMaid \* 1.4.1 2021-10-08 [2] CRAN (R 4.4.1)  
 DBI 1.2.3 2024-06-02 [2] CRAN (R 4.4.0)  
 DEoptimR 1.1-3-1 2024-11-23 [2] CRAN (R 4.4.2)  
 DescTools \* 0.99.58 2024-11-08 [2] CRAN (R 4.4.1)  
 devtools 2.4.5 2022-10-11 [2] CRAN (R 4.4.0)  
 dials \* 1.3.0 2024-07-30 [2] CRAN (R 4.4.1)  
 DiceDesign 1.10 2023-12-07 [2] CRAN (R 4.4.0)  
 digest 0.6.37 2024-08-19 [2] CRAN (R 4.4.1)  
 dplyr \* 1.1.4 2023-11-17 [2] CRAN (R 4.4.0)  
 e1071 1.7-16 2024-09-16 [2] CRAN (R 4.4.1)  
 easypackages 0.1.0 2016-12-05 [2] CRAN (R 4.4.0)  
 ellipsis 0.3.2 2021-04-29 [2] CRAN (R 4.4.0)  
 evaluate 1.0.1 2024-10-10 [2] CRAN (R 4.4.1)  
 Exact 3.3 2024-07-21 [2] CRAN (R 4.4.1)  
 expm 1.0-0 2024-08-19 [2] CRAN (R 4.4.1)  
 extrafont \* 0.19 2023-01-18 [2] CRAN (R 4.4.0)  
 extrafontdb 1.0 2012-06-11 [2] CRAN (R 4.4.0)  
 farver 2.1.2 2024-05-13 [2] CRAN (R 4.4.0)  
 fastmap 1.2.0 2024-05-15 [2] CRAN (R 4.4.0)  
 FAwR \* 1.1.2 2020-11-09 [2] CRAN (R 4.4.0)  
 forcats \* 1.0.0 2023-01-29 [2] CRAN (R 4.4.0)  
 foreach 1.5.2 2022-02-02 [2] CRAN (R 4.4.0)  
 Formula 1.2-5 2023-02-24 [2] CRAN (R 4.4.0)  
 fs 1.6.5 2024-10-30 [2] CRAN (R 4.4.1)  
 furrr 0.3.1 2022-08-15 [2] CRAN (R 4.4.0)  
 future 1.34.0 2024-07-29 [2] CRAN (R 4.4.1)  
 future.apply 1.11.3 2024-10-27 [2] CRAN (R 4.4.1)  
 generics 0.1.3 2022-07-05 [2] CRAN (R 4.4.0)  
 ggplot2 \* 3.5.1 2024-04-23 [2] CRAN (R 4.4.0)  
 gld 2.6.6 2022-10-23 [2] CRAN (R 4.4.0)  
 globals 0.16.3 2024-03-08 [2] CRAN (R 4.4.0)  
 glpkAPI \* 1.3.4 2022-11-10 [2] CRAN (R 4.4.0)  
 glue 1.8.0 2024-09-30 [2] CRAN (R 4.4.1)  
 goftest 1.2-3 2021-10-07 [2] CRAN (R 4.4.0)  
 gower 1.0.2 2024-12-17 [2] CRAN (R 4.4.2)  
 GPfit 1.0-8 2019-02-08 [2] CRAN (R 4.4.0)  
 gridExtra 2.3 2017-09-09 [2] CRAN (R 4.4.0)  
 gtable 0.3.6 2024-10-25 [2] CRAN (R 4.4.1)  
 hardhat 1.4.0 2024-06-02 [2] CRAN (R 4.4.0)  
 haven 2.5.4 2023-11-30 [2] CRAN (R 4.4.0)  
 hms 1.1.3 2023-03-21 [2] CRAN (R 4.4.0)  
 htmltools \* 0.5.8.1 2024-04-04 [2] CRAN (R 4.4.0)  
 htmlwidgets 1.6.4 2023-12-06 [2] CRAN (R 4.4.0)  
 httpuv 1.6.15 2024-03-26 [2] CRAN (R 4.4.0)  
 httr 1.4.7 2023-08-15 [2] CRAN (R 4.4.0)  
 infer \* 1.0.7 2024-03-25 [2] CRAN (R 4.4.0)  
 ipred 0.9-15 2024-07-18 [2] CRAN (R 4.4.1)  
 iterators 1.0.14 2022-02-05 [2] CRAN (R 4.4.0)  
 janitor \* 2.2.0 2023-02-02 [2] CRAN (R 4.4.0)  
 jsonlite \* 1.8.9 2024-09-20 [2] CRAN (R 4.4.1)  
 kableExtra \* 1.4.0 2024-01-24 [2] CRAN (R 4.4.0)  
 kernlab \* 0.9-33 2024-08-13 [2] CRAN (R 4.4.1)  
 KernSmooth 2.23-24 2024-05-17 [2] CRAN (R 4.4.0)  
 knitr \* 1.49 2024-11-08 [2] CRAN (R 4.4.1)  
 labeling 0.4.3 2023-08-29 [2] CRAN (R 4.4.0)  
 later 1.4.1 2024-11-27 [2] CRAN (R 4.4.2)  
 lattice \* 0.22-6 2024-03-20 [2] CRAN (R 4.4.0)  
 lava 1.8.0 2024-03-05 [2] CRAN (R 4.4.0)  
 lazyeval 0.2.2 2019-03-15 [2] CRAN (R 4.4.0)  
 lhs 1.2.0 2024-06-30 [2] CRAN (R 4.4.1)  
 lifecycle 1.0.4 2023-11-07 [2] CRAN (R 4.4.0)  
 listenv 0.9.1 2024-01-29 [2] CRAN (R 4.4.0)  
 lmom 3.2 2024-09-30 [2] CRAN (R 4.4.1)  
 lubridate \* 1.9.4 2024-12-08 [2] CRAN (R 4.4.2)  
 magrittr 2.0.3 2022-03-30 [2] CRAN (R 4.4.0)  
 MASS \* 7.3-61 2024-06-13 [2] CRAN (R 4.4.0)  
 Matrix 1.7-1 2024-10-18 [2] CRAN (R 4.4.1)  
 memoise 2.0.1 2021-11-26 [2] CRAN (R 4.4.0)  
 mime 0.12 2021-09-28 [2] CRAN (R 4.4.0)  
 miniUI 0.1.1.1 2018-05-18 [2] CRAN (R 4.4.0)  
 minpack.lm 1.2-4 2023-09-11 [2] CRAN (R 4.4.0)  
 MLmetrics \* 1.1.3 2024-04-13 [2] CRAN (R 4.4.0)  
 mnormt 2.1.1 2022-09-26 [2] CRAN (R 4.4.0)  
 modeldata \* 1.4.0 2024-06-19 [2] CRAN (R 4.4.0)  
 ModelMetrics 1.2.2.2 2020-03-17 [2] CRAN (R 4.4.0)  
 munsell 0.5.1 2024-04-01 [2] CRAN (R 4.4.0)  
 mvtnorm 1.3-2 2024-11-04 [2] CRAN (R 4.4.1)  
 nlme 3.1-166 2024-08-14 [2] CRAN (R 4.4.1)  
 nnet 7.3-19 2023-05-03 [2] CRAN (R 4.4.0)  
 nortest 1.0-4 2015-07-30 [2] CRAN (R 4.4.0)  
 olsrr \* 0.6.1 2024-11-06 [2] CRAN (R 4.4.1)  
 pander 0.6.5 2022-03-18 [2] CRAN (R 4.4.0)  
 parallelly 1.40.1 2024-12-04 [2] CRAN (R 4.4.2)  
 parsnip \* 1.2.1 2024-03-22 [2] CRAN (R 4.4.0)  
 pillar 1.10.0 2024-12-17 [2] CRAN (R 4.4.2)  
 pkgbuild 1.4.5 2024-10-28 [2] CRAN (R 4.4.1)  
 pkgconfig 2.0.3 2019-09-22 [2] CRAN (R 4.4.0)  
 pkgload 1.4.0 2024-06-28 [2] CRAN (R 4.4.1)  
 plotly \* 4.10.4 2024-01-13 [2] CRAN (R 4.4.0)  
 plyr 1.8.9 2023-10-02 [2] CRAN (R 4.4.0)  
 pROC 1.18.5 2023-11-01 [2] CRAN (R 4.4.0)  
 processx 3.8.4 2024-03-16 [2] CRAN (R 4.4.0)  
 prodlim 2024.06.25 2024-06-24 [2] CRAN (R 4.4.0)  
 profvis 0.4.0 2024-09-20 [2] CRAN (R 4.4.1)  
 promises 1.3.2 2024-11-28 [2] CRAN (R 4.4.2)  
 proxy 0.4-27 2022-06-09 [2] CRAN (R 4.4.0)  
 ps 1.8.1 2024-10-28 [2] CRAN (R 4.4.1)  
 psych \* 2.4.6.26 2024-06-27 [2] CRAN (R 4.4.1)  
 purrr \* 1.0.2 2023-08-10 [2] CRAN (R 4.4.0)  
 R6 2.5.1 2021-08-19 [2] CRAN (R 4.4.0)  
 randomForest 4.7-1.2 2024-09-22 [2] CRAN (R 4.4.1)  
 rappdirs 0.3.3 2021-01-31 [2] CRAN (R 4.4.1)  
 RColorBrewer \* 1.1-3 2022-04-03 [2] CRAN (R 4.4.0)  
 Rcpp 1.0.13-1 2024-11-02 [2] CRAN (R 4.4.1)  
 readr \* 2.1.5 2024-01-10 [2] CRAN (R 4.4.0)  
 readxl \* 1.4.3 2023-07-06 [2] CRAN (R 4.4.0)  
 recipes \* 1.1.0 2024-07-04 [2] CRAN (R 4.4.1)  
 remotes 2.5.0 2024-03-17 [2] CRAN (R 4.4.0)  
 reshape2 1.4.4 2020-04-09 [2] CRAN (R 4.4.0)  
 rlang 1.1.4 2024-06-04 [2] CRAN (R 4.4.0)  
 rmarkdown \* 2.29 2024-11-04 [1] CRAN (R 4.4.2)  
 robustbase 0.99-4-1 2024-09-27 [2] CRAN (R 4.4.1)  
 rootSolve 1.8.2.4 2023-09-21 [2] CRAN (R 4.4.0)  
 rpart 4.1.23 2023-12-05 [2] CRAN (R 4.4.0)  
 rsample \* 1.2.1 2024-03-25 [2] CRAN (R 4.4.0)  
 rstudioapi 0.17.1 2024-10-22 [2] CRAN (R 4.4.1)  
 Rttf2pt1 1.3.12 2023-01-22 [2] CRAN (R 4.4.0)  
 scales \* 1.3.0 2023-11-28 [2] CRAN (R 4.4.0)  
 sessioninfo 1.2.2 2021-12-06 [2] CRAN (R 4.4.0)  
 sf 1.0-19 2024-11-05 [2] CRAN (R 4.4.2)  
 shiny 1.10.0 2024-12-14 [2] CRAN (R 4.4.2)  
 snakecase 0.11.1 2023-08-27 [2] CRAN (R 4.4.0)  
 stringi 1.8.4 2024-05-06 [2] CRAN (R 4.4.0)  
 stringr \* 1.5.1 2023-11-14 [2] CRAN (R 4.4.0)  
 survival 3.8-3 2024-12-17 [2] CRAN (R 4.4.2)  
 svglite 2.1.3 2023-12-08 [2] CRAN (R 4.4.0)  
 systemfonts 1.1.0 2024-05-15 [2] CRAN (R 4.4.0)  
 terra 1.8-5 2024-12-12 [2] CRAN (R 4.4.2)  
 tibble \* 3.2.1 2023-03-20 [2] CRAN (R 4.4.0)  
 tidymodels \* 1.2.0 2024-03-25 [2] CRAN (R 4.4.0)  
 tidyr \* 1.3.1 2024-01-24 [2] CRAN (R 4.4.0)  
 tidyselect 1.2.1 2024-03-11 [2] CRAN (R 4.4.0)  
 tidyverse \* 2.0.0 2023-02-22 [2] CRAN (R 4.4.0)  
 timechange 0.3.0 2024-01-18 [2] CRAN (R 4.4.1)  
 timeDate 4041.110 2024-09-22 [2] CRAN (R 4.4.1)  
 tinytex \* 0.54 2024-11-01 [2] CRAN (R 4.4.1)  
 tune \* 1.2.1 2024-04-18 [2] CRAN (R 4.4.0)  
 tzdb 0.4.0 2023-05-12 [2] CRAN (R 4.4.0)  
 units 0.8-5 2023-11-28 [2] CRAN (R 4.4.0)  
 urlchecker 1.0.1 2021-11-30 [2] CRAN (R 4.4.0)  
 useful \* 1.2.6.1 2023-10-24 [2] CRAN (R 4.4.0)  
 usethis 3.1.0 2024-11-26 [2] CRAN (R 4.4.2)  
 utf8 1.2.4 2023-10-22 [2] CRAN (R 4.4.0)  
 vctrs 0.6.5 2023-12-01 [2] CRAN (R 4.4.0)  
 viridisLite 0.4.2 2023-05-02 [2] CRAN (R 4.4.0)  
 webshot \* 0.5.5 2023-06-26 [2] CRAN (R 4.4.0)  
 webshot2 \* 0.1.1 2023-08-11 [2] CRAN (R 4.4.0)  
 websocket 1.4.2 2024-07-22 [2] CRAN (R 4.4.1)  
 withr 3.0.2 2024-10-28 [2] CRAN (R 4.4.1)  
 workflows \* 1.1.4 2024-02-19 [2] CRAN (R 4.4.0)  
 workflowsets \* 1.1.0 2024-03-21 [2] CRAN (R 4.4.0)  
 xfun 0.49 2024-10-31 [2] CRAN (R 4.4.1)  
 xml2 1.3.6 2023-12-04 [2] CRAN (R 4.4.0)  
 xtable 1.8-4 2019-04-21 [2] CRAN (R 4.4.0)  
 yaml 2.3.10 2024-07-26 [2] CRAN (R 4.4.1)  
 yardstick \* 1.3.1 2024-03-21 [2] CRAN (R 4.4.0)  
  
 [1] /home/seamus/R/x86\_64-redhat-linux-gnu-library/4.4  
 [2] /usr/local/lib/R/library  
 [3] /usr/lib64/R/library  
 [4] /usr/share/R/library  
  
──────────────────────────────────────────────────────────────────────────────