

# Production Code: ‘DEM-based Pipeline (Including Developed Blocks)’

Cabin-GIS

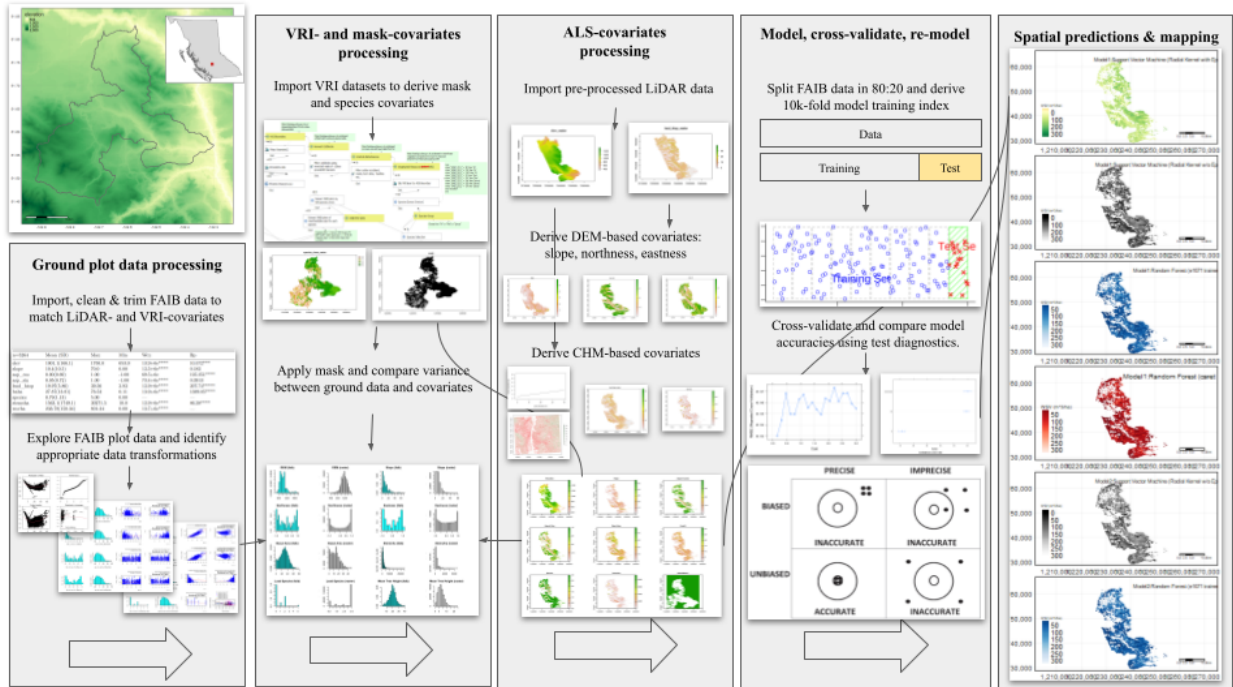
14/06/2022

## Contents

Action . . . . .	1
Import: Load covariates . . . . .	2
Import: Load mask layers . . . . .	2
Tidy: Align, clip and apply masking . . . . .	4
Tidy: Merge , rename, stack covariates . . . . .	5
Tidy: format and bootstrap training data . . . . .	7
Model: Random Forest Regression Tree . . . . .	8

## Action

The following markdown report provides a complete run-through and guide of a raster-to-raster workflow to generating Whole Stem Volume ( $\text{m}^3/\text{ha}$ : WSVHA) raster estimates from stage of importing and masking DEM-based and species covariates, to fitting and training models with faib.csv data, to finally making spatial predictions using raster stack of covariates. The graphical abstract below is offered as reference guide.



## Import: Load covariates

```
elev_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates/d
elev_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates/d
slope_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates/
slope_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates/
asp_cos_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates
asp_cos_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates
asp_sin_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates
asp_sin_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariates
species_class_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-cov
species_class_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-cov
lead_htop_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covaria
lead_htop_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covaria
#masks_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mask/mask_raster_10
#masks_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mask/mask_raster_10
```

## Import: Load mask layers

```
mask_burn2017 = sf::read_sf("/media/seamus/128GB_WORKD/data/vector/tcc_mask_layers/TCC_Burn_Severity TCC
mask_burn2018 = sf::read_sf("/media/seamus/128GB_WORKD/data/vector/tcc_mask_layers/TCC_Burn_Severity TCC
mask_burn2021 = sf::read_sf("/media/seamus/128GB_WORKD/data/vector/tcc_mask_layers/TCC_Burn_Severity TCC
mask_burn2017 = mask_burn2017["BurnSev"]
mask_burn2018 = mask_burn2018["BurnSev"]
mask_burn2021 = mask_burn2021["BurnSev"]
mask_burn2017 = dplyr::filter(mask_burn2017, BurnSev == 'High')
mask_burn2018 = dplyr::filter(mask_burn2018, BurnSev == 'High')
```

```

mask_burn2021 = dplyr::filter(mask_burn2021, BurnSev == 'High')
lead_htop_sv_quesnel = as.polygons(lead_htop_rast_quesnel)
lead_htop_sv_gaspard = as.polygons(lead_htop_rast_gaspard)
lead_htop_sf_quesnel = sf::st_as_sf(lead_htop_sv_quesnel)
lead_htop_sf_gaspard = sf::st_as_sf(lead_htop_sv_gaspard)
mask_burn2017_quesnel = sf::st_intersection(sf::st_make_valid(mask_burn2017), lead_htop_sf_quesnel)
mask_burn2017_gaspard = sf::st_intersection(sf::st_make_valid(mask_burn2017), lead_htop_sf_gaspard)
mask_burn2018_quesnel = sf::st_intersection(sf::st_make_valid(mask_burn2018), lead_htop_sf_quesnel)
mask_burn2018_gaspard = sf::st_intersection(sf::st_make_valid(mask_burn2018), lead_htop_sf_gaspard)
mask_burn2021_quesnel = sf::st_intersection(sf::st_make_valid(mask_burn2021), lead_htop_sf_quesnel)
mask_burn2021_gaspard = sf::st_intersection(sf::st_make_valid(mask_burn2021), lead_htop_sf_gaspard)
masks_df_quesnel = full_join(as_tibble(mask_burn2017_quesnel), as_tibble(mask_burn2018_quesnel), as_tibble(mask_burn2021_quesnel))
masks_df_gaspard = full_join(as_tibble(mask_burn2017_gaspard), as_tibble(mask_burn2018_gaspard), as_tibble(mask_burn2021_gaspard))
masks_sf_quesnel = st_as_sf(masks_df_quesnel) # easier to combine by 'geometry'
masks_sf_gaspard = st_as_sf(masks_df_gaspard) # easier to combine by 'geometry'

#mask_clearcut = sf::read_sf("/media/seamus/128GB_WORKKD/data/vector/tcc_mask_layers/RSLT_CCRES_CLEAR.shp")
#mask_clearcut_quesnel = sf::st_intersection(mask_clearcut, st_make_valid(lead_htop_sf_quesnel))
#mask_clearcut_gaspard = sf::st_intersection(mask_clearcut, st_make_valid(lead_htop_sf_gaspard))
#masks_df_quesnel = full_join(as_tibble(masks_sf_quesnel), as_tibble(mask_clearcut_quesnel), by = 'geometry')
#masks_df_gaspard = full_join(as_tibble(masks_sf_gaspard), as_tibble(mask_clearcut_gaspard), by = 'geometry')
#masks_sf_quesnel = st_as_sf(masks_df_quesnel)
#masks_sf_gaspard = st_as_sf(masks_df_gaspard)

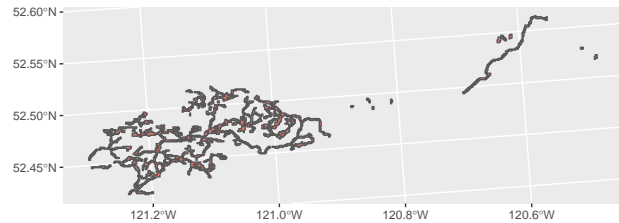
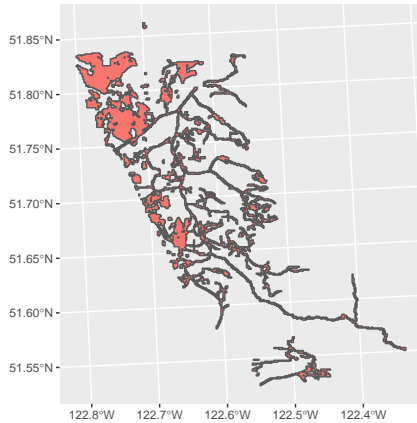
#mask_blocks = sf::read_sf("/media/seamus/128GB_WORKKD/data/vector/tcc_mask_layers/TCC_Blocks_Join.shp")
#mask_blocks_quesnel = sf::st_intersection(mask_blocks, st_make_valid(lead_htop_sf_quesnel))
#mask_blocks_gaspard = sf::st_intersection(mask_blocks, st_make_valid(lead_htop_sf_gaspard))
#masks_df_quesnel = full_join(as_tibble(masks_sf_quesnel), as_tibble(mask_blocks_quesnel), by = 'geometry')
#masks_df_gaspard = full_join(as_tibble(masks_sf_gaspard), as_tibble(mask_blocks_gaspard), by = 'geometry')
#masks_sf_quesnel = st_as_sf(masks_df_quesnel)
#masks_sf_gaspard = st_as_sf(masks_df_gaspard)

mask_roads_tcc = sf::read_sf("/media/seamus/128GB_WORKKD/data/vector/tcc_mask_layers/TCC_Roads.shp")
mask_roads_tcc = sf::st_zm(mask_roads_tcc)
mask_roads_tcc_quesnel = sf::st_intersection(mask_roads_tcc, st_make_valid(lead_htop_sf_quesnel))
mask_roads_tcc_gaspard = sf::st_intersection(mask_roads_tcc, st_make_valid(lead_htop_sf_gaspard))
mask_roads_tcc_quesnel = sf::st_buffer(mask_roads_tcc_quesnel, dist = 15, nQuadSegs = 5, endCapStyle = 1)
mask_roads_tcc_gaspard = sf::st_buffer(mask_roads_tcc_gaspard, dist = 15, nQuadSegs = 5, endCapStyle = 1)
mask_roads_ften = sf::read_sf("/media/seamus/128GB_WORKKD/data/vector/tcc_mask_layers/FTEN_Roads_All.shp")
mask_roads_ften = sf::st_zm(mask_roads_ften)
mask_roads_ften_quesnel = sf::st_intersection(mask_roads_ften, st_make_valid(lead_htop_sf_quesnel))
mask_roads_ften_gaspard = sf::st_intersection(mask_roads_ften, st_make_valid(lead_htop_sf_gaspard))
mask_roads_ften_quesnel = sf::st_buffer(mask_roads_ften_quesnel, dist = 15, nQuadSegs = 5, endCapStyle = 1)
mask_roads_ften_gaspard = sf::st_buffer(mask_roads_ften_gaspard, dist = 15, nQuadSegs = 5, endCapStyle = 1)
masks_df_quesnel = full_join(as_tibble(masks_sf_quesnel), as_tibble(mask_roads_tcc_quesnel), as_tibble(mask_roads_ften_quesnel))
masks_df_gaspard = full_join(as_tibble(masks_sf_gaspard), as_tibble(mask_roads_tcc_gaspard), as_tibble(mask_roads_ften_gaspard))
masks_sf_quesnel = st_as_sf(masks_df_quesnel)
masks_sf_gaspard = st_as_sf(masks_df_gaspard)

masks_rast_quesnel = rasterize(vect(masks_sf_quesnel), lead_htop_rast_quesnel, touches = TRUE)
masks_rast_gaspard = rasterize(vect(masks_sf_gaspard), lead_htop_rast_gaspard, touches = TRUE)
masks_raster_quesnel = raster::raster(masks_rast_quesnel)
masks_raster_gaspard = raster::raster(masks_rast_gaspard)

```

```
writeRaster(masks_raster_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mask/mas
writeRaster(masks_raster_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mask/mas
ggplot(masks_sf_gaspard) + geom_sf(aes(fill = 'red'), show.legend = FALSE)
ggplot(masks_sf_quesnel) + geom_sf(aes(fill = 'red'), show.legend = FALSE)
```



## Tidy: Align, clip and apply masking

```
# masking by mask
masks_rast_quesnel = terra::resample(masks_rast_quesnel, lead_htop_rast_quesnel)
masks_rast_gaspard = terra::resample(masks_rast_gaspard, lead_htop_rast_gaspard)
masks_rast_quesnel = terra::resample(masks_rast_quesnel, elev_rast_quesnel)
masks_rast_gaspard = terra::resample(masks_rast_gaspard, elev_rast_gaspard)
lead_htop_rast_quesnel = mask(lead_htop_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
lead_htop_rast_gaspard = mask(lead_htop_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
elev_rast_quesnel = mask(elev_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
elev_rast_gaspard = mask(elev_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
slope_rast_quesnel = mask(slope_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
slope_rast_gaspard = mask(slope_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
asp_cos_rast_quesnel = mask(asp_cos_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
asp_cos_rast_gaspard = mask(asp_cos_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
asp_sin_rast_quesnel = mask(asp_sin_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
asp_sin_rast_gaspard = mask(asp_sin_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
species_class_rast_quesnel = mask(species_class_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
species_class_rast_gaspard = mask(species_class_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
# masking by species
lead_htop_rast_quesnel = mask(lead_htop_rast_quesnel, species_class_rast_quesnel, inverse=FALSE)
lead_htop_rast_gaspard = mask(lead_htop_rast_gaspard, species_class_rast_gaspard, inverse=FALSE)
elev_rast_quesnel = mask(elev_rast_quesnel, species_class_rast_quesnel, inverse=FALSE)
elev_rast_gaspard = mask(elev_rast_gaspard, species_class_rast_gaspard, inverse=FALSE)
slope_rast_quesnel = mask(slope_rast_quesnel, species_class_rast_quesnel, inverse=FALSE)
slope_rast_gaspard = mask(slope_rast_gaspard, species_class_rast_gaspard, inverse=FALSE)
asp_cos_rast_quesnel = mask(asp_cos_rast_quesnel, species_class_rast_quesnel, inverse=FALSE)
asp_cos_rast_gaspard = mask(asp_cos_rast_gaspard, species_class_rast_gaspard, inverse=FALSE)
asp_sin_rast_quesnel = mask(asp_sin_rast_quesnel, species_class_rast_quesnel, inverse=FALSE)
asp_sin_rast_gaspard = mask(asp_sin_rast_gaspard, species_class_rast_gaspard, inverse=FALSE)
# save outputs
writeRaster(elev_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-covar
```

```

writeRaster(elev_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-cova
writeRaster(slope_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-cova
writeRaster(slope_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-cova
writeRaster(asp_cos_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-c
writeRaster(asp_cos_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-c
writeRaster(asp_sin_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-c
writeRaster(asp_sin_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-c
writeRaster(species_class_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/ma
writeRaster(species_class_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/ma
writeRaster(lead_htop_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked
writeRaster(lead_htop_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked

```

## Tidy: Merge , rename, stack covariates

```

# transform spatRaster to raster
elev_raster_quesnel = raster::raster(elev_rast_quesnel)
slope_raster_quesnel = raster::raster(slope_rast_quesnel)
asp_cos_raster_quesnel = raster::raster(asp_cos_rast_quesnel)
asp_sin_raster_quesnel = raster::raster(asp_sin_rast_quesnel)
species_class_raster_quesnel = raster::raster(species_class_rast_quesnel)
lead_htop_raster_quesnel = raster::raster(lead_htop_rast_quesnel)

elev_raster_gaspard = raster::raster(elev_rast_gaspard)
slope_raster_gaspard = raster::raster(slope_rast_gaspard)
asp_cos_raster_gaspard = raster::raster(asp_cos_rast_gaspard)
asp_sin_raster_gaspard = raster::raster(asp_sin_rast_gaspard)
species_class_raster_gaspard = raster::raster(species_class_rast_gaspard)
lead_htop_raster_gaspard = raster::raster(lead_htop_rast_gaspard)

# merge AllAreas rasters
elev_raster_list = list(elev_raster_quesnel, elev_raster_gaspard)
slope_raster_list = list(slope_raster_quesnel, slope_raster_gaspard)
asp_cos_raster_list = list(asp_cos_raster_quesnel, asp_cos_raster_gaspard)
asp_sin_raster_list = list(asp_sin_raster_quesnel, asp_sin_raster_gaspard)
species_class_raster_list = list(species_class_raster_quesnel, species_class_raster_gaspard)
lead_htop_raster_list = list(lead_htop_raster_quesnel, lead_htop_raster_gaspard)

elev_raster = do.call(merge, c(elev_raster_list, tolerance = 1))
slope_raster = do.call(merge, c(slope_raster_list, tolerance = 1))
asp_cos_raster = do.call(merge, c(asp_cos_raster_list, tolerance = 1))
asp_sin_raster = do.call(merge, c(asp_sin_raster_list, tolerance = 1))
species_class_raster = do.call(merge, c(species_class_raster_list, tolerance = 1))
lead_htop_raster = do.call(merge, c(lead_htop_raster_list, tolerance = 1))

# Rename rasters
names(elev_rast_quesnel) = "elev"
names(slope_rast_quesnel) = "slope"
names(asp_cos_rast_quesnel) = "asp_cos"
names(asp_sin_rast_quesnel) = "asp_sin"
names(species_class_rast_quesnel) = "species_class"
names(lead_htop_rast_quesnel) = "lead_htop"

names(elev_rast_gaspard) = "elev"

```

```

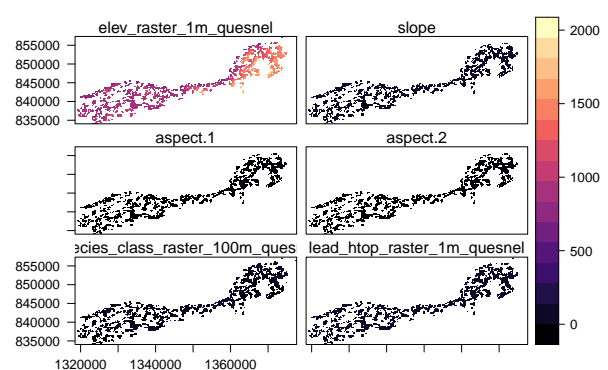
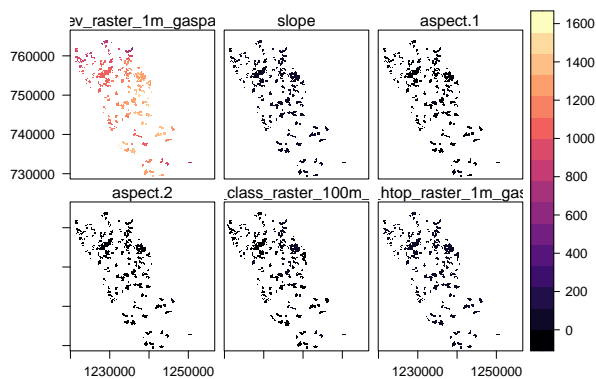
names(slope_rast_gaspard) = "slope"
names(asp_cos_rast_gaspard) = "asp_cos"
names(asp_sin_rast_gaspard) = "asp_sin"
names(species_class_rast_gaspard) = "species_class"
names(lead_htop_rast_gaspard) = "lead_htop"

names(elev_raster) = "elev"
names(slope_raster) = "slope"
names(asp_cos_raster) = "asp_cos"
names(asp_sin_raster) = "asp_sin"
names(species_class_raster) = "species_class"
names(lead_htop_raster) = "lead_htop"
# stack rasters
covs_m1_quesnel = raster::stack(
  elev_raster_quesnel,
  slope_raster_quesnel,
  asp_cos_raster_quesnel,
  asp_sin_raster_quesnel,
  species_class_raster_quesnel,
  lead_htop_raster_quesnel)

covs_m1_gaspard = raster::stack(
  elev_raster_gaspard,
  slope_raster_gaspard,
  asp_cos_raster_gaspard,
  asp_sin_raster_gaspard,
  species_class_raster_gaspard,
  lead_htop_raster_gaspard)

covs_m1 = raster::stack(
  elev_raster,
  slope_raster,
  asp_cos_raster,
  asp_sin_raster,
  lead_htop_raster,
  species_class_raster)
# visualize
rasterVis::levelplot(covs_m1_gaspard)
rasterVis::levelplot(covs_m1_quesnel)

```





## Tidy: format and bootstrap training data

```
faib_psp <- read.csv("/media/seamus/128GB_WORKD/EFI-TCC/0_Caret_Predict_to_writeRasterOutput/Data/FAIB_L")
faib_psp = subset(faib_psp, util == '12.5')
faib_psp$spc_live1 = as.factor(faib_psp$spc_live1)
faib_psp = subset(
  faib_psp, spc_live1=='PL' | spc_live1=='PLI' | spc_live1=='FD' | spc_live1=='FDI' |
  spc_live1=='SB' | spc_live1=='SE' | spc_live1=='SW' | spc_live1=='SX' |
  spc_live1=='CW' | spc_live1=='HW' | spc_live1=='BL' | spc_live1=='LW')
faib_psp$species_class = dplyr::recode(
  faib_psp$spc_live1, PL = 1, PLI = 1, SB = 2, SE = 2, SX = 2,
  FD = 3, FDI = 3, CW = 3, HW = 4, BL = 5, LW = 6)
faib_psp$asp_cos = cos((faib_psp$aspect * pi) / 180)
faib_psp$asp_sin = sin((faib_psp$aspect * pi) / 180)

faib_psp$elev[faib_psp$elev <= 0] = NA
faib_psp$slope[faib_psp$slope <= 0] = NA
faib_psp$lead_htop[faib_psp$lead_htop < 2] = NA
faib_psp$stemsha_L[faib_psp$stemsha_L <= 0] = NA
faib_psp$wsvha_L[faib_psp$wsvha_L <= 0] = NA
faib_psp = subset(faib_psp, stemsha_L < 864)

faib_psp$elev = as.numeric(faib_psp$elev)
faib_psp$slope = as.numeric(faib_psp$slope)
faib_psp$asp_cos = as.numeric(faib_psp$asp_cos)
faib_psp$asp_sin = as.numeric(faib_psp$asp_sin)
faib_psp$lead_htop = as.numeric(faib_psp$lead_htop)
faib_psp$species_class = as.numeric(faib_psp$species_class)
faib_psp$stemsha_L = as.numeric(faib_psp$stemsha_L)
faib_psp$wsvha_L = as.numeric(faib_psp$wsvha_L)
faib_vri_true_m1_df = faib_psp[c("elev", "slope", "asp_cos", "asp_sin", "lead_htop", "species_class", "wsvha_L")]
faib_vri_true_m1_df = na.omit(faib_vri_true_m1_df)

lead_htop_raster_df = as.data.frame(rasterToPoints(lead_htop_raster))
dens.fun = approxfun(density(lead_htop_raster_df$lead_htop, adjust=0.8))
B = 1000
n = 4
faib_vri_true_m1_df_boot = dplyr::sample_n(faib_vri_true_m1_df, B * n, weight_by = dens.fun(faib_vri_true_m1_df$lead_htop))
faib_vri_true_m1_df_boot = na.omit(faib_vri_true_m1_df_boot)
truehist(faib_vri_true_m1_df$lead_htop, main="CHM (un-Bootstrapped FAIB)")
truehist(faib_vri_true_m1_df_boot$lead_htop, main="CHM (Bootstrapped FAIB)")

faib_vri_true_m1_df_boot_split = createDataPartition(faib_vri_true_m1_df_boot$wsvha_L, p=0.80, list=F)
faib_vri_true_m1_df_split = createDataPartition(faib_vri_true_m1_df$wsvha_L, p=0.80, list=F)

train_m1_boot = faib_vri_true_m1_df_boot[faib_vri_true_m1_df_boot_split, ]
test_m1_boot = faib_vri_true_m1_df_boot[-faib_vri_true_m1_df_boot_split, ]
train_m1 = faib_vri_true_m1_df[faib_vri_true_m1_df_split, ]
test_m1 = faib_vri_true_m1_df[-faib_vri_true_m1_df_split, ]

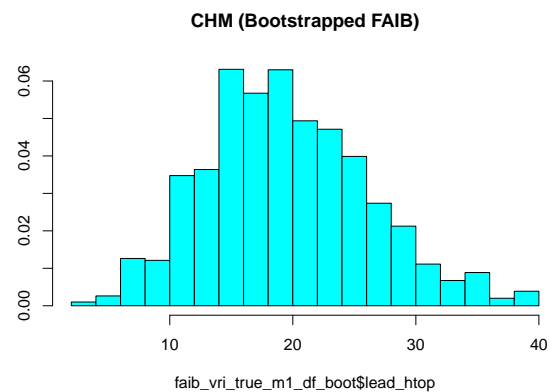
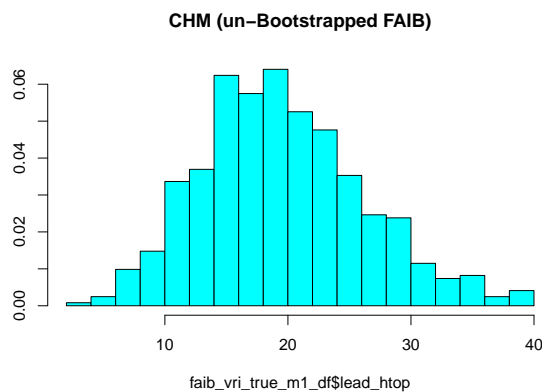
X_train_m1_boot = train_m1_boot[, -7]
y_train_m1_boot = train_m1_boot[, 7]
X_test_m1_boot = test_m1_boot[, -7]
```

```

y_test_m1_boot = test_m1_boot[, 7]
X_m1_boot = faib_vri_true_m1_df_boot[, -7]
y_m1_boot = faib_vri_true_m1_df_boot[, 7]

X_train_m1 = train_m1[, -7]
y_train_m1 = train_m1[, 7]
X_test_m1 = test_m1[, -7]
y_test_m1 = test_m1[, 7]
X_m1 = faib_vri_true_m1_df[, -7]
y_m1 = faib_vri_true_m1_df[, 7]

```



## Model: Random Forest Regression Tree

```

tuneResult_rf_m1_full <- tune.randomForest(
  X_m1_boot, y_m1_boot,
  mtry = c(2:10), ntree = 50,
  tunecontrol = tune.control(sampling = "cross", cross = 10),
  preProcess = c('YeoJohnson', 'scale', 'center', 'corr'))

tuneResult_rf_m1_train <- tune.randomForest(
  X_train_m1_boot, y_train_m1_boot,
  mtry = c(2:10), ntree = 50,
  tunecontrol = tune.control(sampling = "cross", cross = 10),
  preProcess = c('YeoJohnson', 'scale', 'center', 'corr'))
# Predict models
tunedModel_rf_m1_full <- tuneResult_rf_m1_full$best.model
tunedModel_rf_m1_train <- tuneResult_rf_m1_train$best.model
tunedModel_rf_m1 = predict(tunedModel_rf_m1_full, newdata=faib_vri_true_m1_df, type = "response")
tunedModel_rf_m1_test = predict(tunedModel_rf_m1_train, newdata=test_m1, type = "response")
save(tunedModel_rf_m1_full, file = "/media/seamus/128GB_WORKD/data/models/tcc-wsvha/wsvha_model1_random")
# Performance metrics
tuneResult_rf_m1_full
R2(tunedModel_rf_m1, faib_vri_true_m1_df$wsvha_L)
MAE(tunedModel_rf_m1, faib_vri_true_m1_df$wsvha_L)
RMSE(tunedModel_rf_m1, faib_vri_true_m1_df$wsvha_L)
MAE(tunedModel_rf_m1_test, test_m1$wsvha_L)
RMSE(tunedModel_rf_m1_test, test_m1$wsvha_L)

```



# ``` # Predict rasters ```

```
wsvha_model1_randomForest_bootstrapped_demBased_100m_allAreas <- raster::predict(covs_m1, tunedModel_rf,
wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard <- raster::predict(covs_m1_gaspard, tunedM
wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel <- raster::predict(covs_m1_quesnel, tunedM
wsvha_model1_randomForest_bootstrapped_demBased_100m_allAreas$layer[wsvha_model1_randomForest_bootstrap
wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard$layer[wsvha_model1_randomForest_bootstrap
wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel$layer[wsvha_model1_randomForest_bootstrap
```

# ``` # Save outputs ```

```
writeRaster(wsvha_model1_randomForest_bootstrapped_demBased_100m_allAreas, overwrite=TRUE,
  filename = "/media/seamus/128GB_WORKD/data/raster/tcc/outputs/wsvha/bootstrapped/wsvha_model1_randomF
writeRaster(wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard, overwrite=TRUE,
  filename = "/media/seamus/128GB_WORKD/data/raster/tcc/outputs/wsvha/bootstrapped/wsvha_model1_randomF
writeRaster(wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel, overwrite=TRUE,
  filename = "/media/seamus/128GB_WORKD/data/raster/tcc/outputs/wsvha/bootstrapped/wsvha_model1_randomF
```

# ``` # Visualize outputs ```

```
plot(wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard, main="Gaspard WSVHA: Random Forest",
plot(wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel, main="Quesnel WSVHA: Random Forest",
hist(wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard, main="Gaspard WSVHA: Random Forest",
hist(wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel, main="Quesnel WSVHA: Random Forest",
```

