Production Code: '0_LiDAR-FAIB-WSVHA-raster-to-raster-production-v3.R'

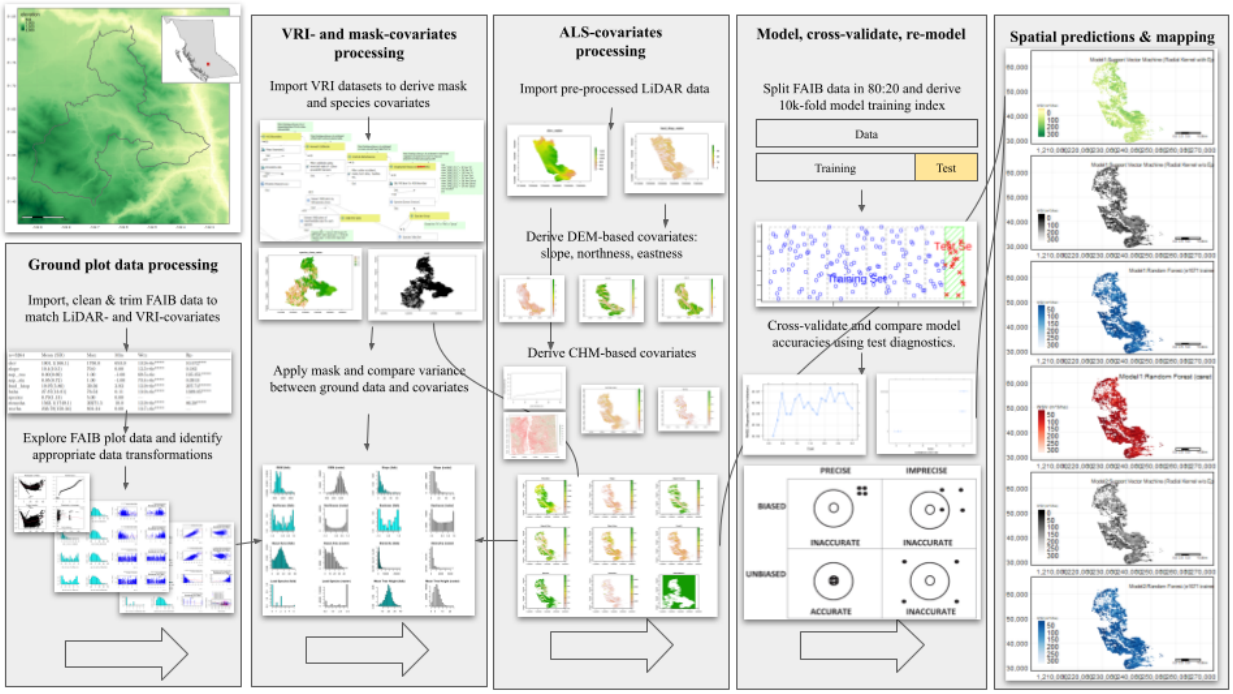
Cabin-GIS

09/06/2022

Contents

Action

The following markdown report provides a complete run-through and guide of a raster-to-raster workflow to generating Whole Stem Volume (m^3/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 and mask layer

```
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-covariate
asp_cos_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariate
asp_sin_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariate
asp_sin_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariate
species_class_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-cova
species_class_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-cova
lead_htop_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariat
lead_htop_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/unmasked-covariat
masks_rast_quesnel = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mask/mask_raster_100
masks_rast_gaspard = terra::rast("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mask/mask_raster_100
```

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)
```

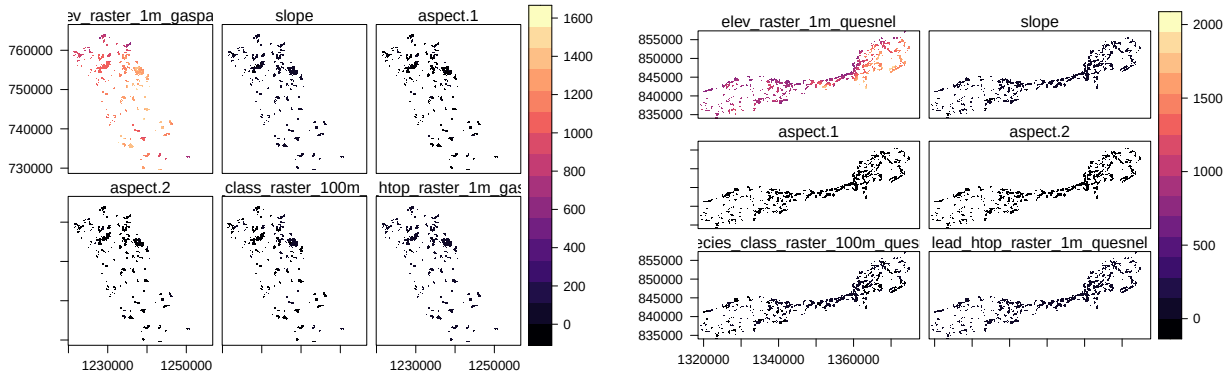
```
elev_rast_gaspard = mask(elev_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
slope_rast_quesnel = mask(slope_rast_quesnel, masks_rast_quesnel, inverse=TRUE)
slope_rast_gaspard = mask(slope_rast_gaspard, masks_rast_gaspard, inverse=TRUE)
asp_cos_rast_quesnel = mask(asp_cos_rast_quesnel, masks_rast_quesnel, 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-covar
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-co
writeRaster(asp_cos_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-co
writeRaster(asp_sin_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-co
writeRaster(asp_sin_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-co
writeRaster(species_class_rast_quesnel, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mas
writeRaster(species_class_rast_gaspard, filename = "/media/seamus/128GB_WORKD/data/raster/tcc/inputs/mas
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,
```

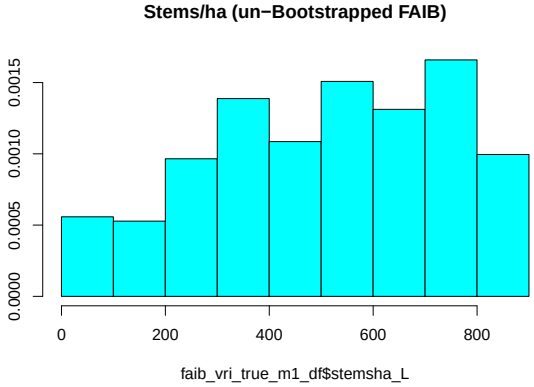
```
  slope_raster,
  asp_cos_raster,
  asp_sin_raster,
  lead_htop_raster,
  species_class_raster)
# visualize
rasterVis::levelplot(covs_m1_gaspard)
rasterVis::levelplot(covs_m1_quesnel)
```

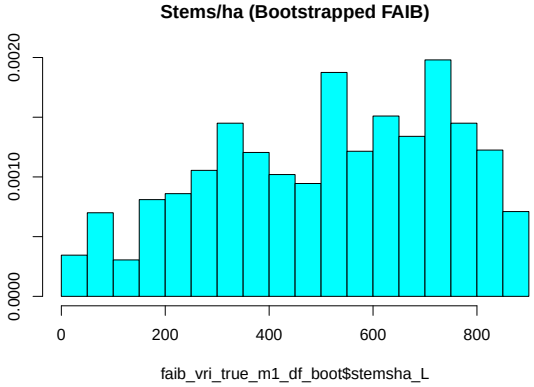


Tidy: format and bootstrap training data

```
faib_psp <- read.csv("/media/seamus/128GB_WORKD/EFI-TCC/0_Caret_Predict_to_writeRasterOutput/Data/FAIB_
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)
```

```r
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", "

stemsha_L_raster = raster::raster("/media/seamus/128GB_WORKD/data/raster/tcc/inputs/masked-covariates/cl
stemsha_L_raster_df = as.data.frame(rasterToPoints(stemsha_L_raster))
dens.fun = approxfun(density(stemsha_L_raster_df$stemsha_L, adjust=0.8))
B = 1000
n = 4
faib_vri_true_m1_df_boot = dplyr::sample_n(faib_vri_true_m1_df, B * n, weight_by = dist.fun(faib_vri_tru
truehist(faib_vri_true_m1_df$stemsha_L, main="Stems/ha (un-Bootstrapped FAIB)")
truehist(faib_vri_true_m1_df_boot$stemsha_L, main="Stems/ha (Bootstrapped FAIB)")

faib_vri_true_m1_df_boot_stemfree = faib_vri_true_m1_df_boot[c("elev", "slope", "asp_cos", "asp_sin", "
faib_vri_true_m1_df_boot_stemfree = na.omit(faib_vri_true_m1_df_boot_stemfree)
faib_vri_true_m1_df_boot_stemfree_split = createDataPartition(faib_vri_true_m1_df_boot_stemfree$wsvha_L
train_m1_boot = faib_vri_true_m1_df_boot_stemfree[faib_vri_true_m1_df_boot_stemfree_split, ]
test_m1_boot = faib_vri_true_m1_df_boot_stemfree[-faib_vri_true_m1_df_boot_stemfree_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_stemfree[,-7]
y_m1_boot = faib_vri_true_m1_df_boot_stemfree[, 7]

faib_vri_true_m1_df = faib_psp[c("elev", "slope", "asp_cos", "asp_sin", "lead_htop", "species_class", "
faib_vri_true_m1_df = na.omit(faib_vri_true_m1_df)
faib_vri_true_m1_df_split = createDataPartition(faib_vri_true_m1_df$wsvha_L, p=0.80, list=F)
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 = 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_randoml
# 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_bootstrapp
wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard$layer[wsvha_model1_randomForest_bootstrappe
wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel$layer[wsvha_model1_randomForest_bootstrappe
# Save outputs
writeRaster(wsvha_model1_randomForest_bootstrapped_demBased_100m_allAreas, overwrite=TRUE,
  filename = "/media/seamus/128GB_WORKD/data/raster/tcc/outputs/wsvha/wsvha_model1_randomForest_bootstra
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",
```

“‘{r, fig.show=‘hold’, out.width=“25%”, echo=FALSE} wsvha_model1_randomForest_bootstrapped_demBased_100m_ga
= raster::raster(“/media/seamus/128GB_WORKD/data/raster/tcc/outputs/wsvha/bootstrapped/wsvha_model1_random
wsvha_model1_randomForest_bootstrapped_demBased_100m_quesnel = raster::raster(“/media/seamus/128GB_WORKI
plot(wsvha_model1_randomForest_bootstrapped_demBased_100m_gaspard, main=“Gaspard WSVHA:
Random Forest”, cex.main=0.8, maxpixels=22000000) plot(wsvha_model1_randomForest_bootstrapped_demBased_100m_
main=“Quesnel WSVHA: Random Forest”, cex.main=0.8, maxpixels=22000000) hist(wsvha_model1_randomForest_bootstr
main=“Gaspard WSVHA: Random Forest”, cex.main=0.8, maxpixels=22000000) hist(wsvha_model1_randomForest_bootst

main="Quesnel WSVHA: Random Forest", cex.main=0.8, maxpixels=22000000)