Untitled

Luke Wertis

2022-10-18

Read in the data

This dataset contains information extracted from the LiDAR dataset and the aerial imagery data. All variables are being looked at initially to try and predict the type of tree being observed.

```
Data <- fread("C:\\Users\\wertisml\\Documents\\Hemlocks\\Scripts_and_Tools\\tree_crowns\\Final\\csv\\Tr

n.cores <- parallel::detectCores() - 1
```

Prepare the data

zentropy has many NA values within its column and for that reason it is being removed from the analysis. Additionally, for a random forest classification to work, the outcome variable, Tree_Type, needs to be a factor.

The data is then partined using a 80/20 train/test split. The train will be used for tuning the random forest model and the test will be left out of the training and will be used at the end to test how well the model performs when predicting on new data.

```
# Fix the names of the data if needed
Data <- Data %>%
  dplyr::select(-treeID, -zentrpy, -ipcmz10, -ipcmz30, -ipcmz50, -ipcmz70, -ipcmz90,
                -treID_3, -imean_1, -isd_1, -iskew_1, -ikurt_1) %>%
  rename(ipcmz10 = ipc10_1,
         ipcmz30 = ipc30_1,
         ipcmz50 = ipc50_1,
         ipcmz70 = ipc70_1,
         ipcmz90 = ipc90_1)

names(Data)
```

```
##  [1] "Z"          "npoints"    "cnvhll_"    "zmax"       "zmean"
##  [6] "zsd"        "zskew"      "zkurt"      "pzbvzmn"    "pzabov2"
## [11] "zq5"        "zq10"       "zq15"       "zq20"       "zq25"
## [16] "zq30"       "zq35"       "zq40"       "zq45"       "zq50"
## [21] "zq55"       "zq60"       "zq65"       "zq70"       "zq75"
## [26] "zq80"       "zq85"       "zq90"       "zq95"       "zpcum1"
## [31] "zpcum2"     "zpcum3"     "zpcum4"     "zpcum5"     "zpcum6"
## [36] "zpcum7"     "zpcum8"     "zpcum9"     "itot"       "imax"
## [41] "imean"      "isd"        "iskew"      "ikurt"      "ipgrond"
## [46] "p1th"       "p2th"       "p3th"       "p4th"       "p5th"
```

```
## [51] "pground"     "n"           "area"        "egn_lrg"     "egn_mdm"
## [56] "egn_sml"     "curvatr"     "linerty"     "planrty"     "sphrcty"
## [61] "anstrpy"     "hrzntlt"     "ipcmz05"     "ipcmz10"     "ipcmz15"
## [66] "ipcmz20"     "ipcmz25"     "ipcmz30"     "ipcmz35"     "ipcmz40"
## [71] "ipcmz45"     "ipcmz50"     "ipcmz55"     "ipcmz60"     "ipcmz65"
## [76] "ipcmz70"     "ipcmz75"     "ipcmz80"     "ipcmz85"     "ipcmz90"
## [81] "ipcmz95"     "ipcm100"     "R"           "G"           "B"
## [86] "Canopy_Base" "Tree_Type"
```

```
# Remove the NA rows
Data <- Data[complete.cases(Data),]

Data$Tree_Type <- as.factor(Data$Tree_Type)

set.seed(24)
split <- rsample::initial_split(Data,
                                prop = 0.7,
                                strata = Tree_Type)

Train <- training(split)
Test <- testing(split)

# Preprocessing "recipe"
preprocessing_recipe <- recipes::recipe(Tree_Type ~ ., data = Train) %>%
  # convert categorical variables to factors (One hot encoding)
  recipes::step_string2factor(all_nominal()) %>%
  # combine low frequency factor levels
  recipes::step_other(all_nominal(), threshold = 0.01) %>%
  # remove no variance predictors which provide no predictive information
  recipes::step_nzv(all_nominal())

tree_prep <- prep(preprocessing_recipe)
juiced <- juice(tree_prep)
```

Model Tuning

a grid is created to create multiple combinations of models (250) for the random forest to test. The best model will have the lowest prediction error.

Additionally, this model is run in parallel to allow for faster computation. This is not necessary for this model at the moment, but if a lot of new information was added it would be useful to already have set up.

```
# Cross fold validation
set.seed(24)
trees_folds <- vfold_cv(Train, v = 10)

# Build the model frame
tune_spec <- rand_forest(mtry = tune(),
                         trees = tune(),
                         min_n = tune()) %>%
  set_mode("classification") %>%
  set_engine("ranger")
```

```
# Set up the workflow for the model to follow
tune_wf <- workflow() %>%
  add_recipe(preprocessing_recipe) %>%
  add_model(tune_spec)

#create and register cluster
my.cluster <- parallel::makeCluster(n.cores)
doParallel::registerDoParallel(cl = my.cluster)

# Tune the hyperparamters
set.seed(24)
tune_res <- tune_grid(tune_wf,
                      resamples = trees_folds,
                      grid = 60)
```

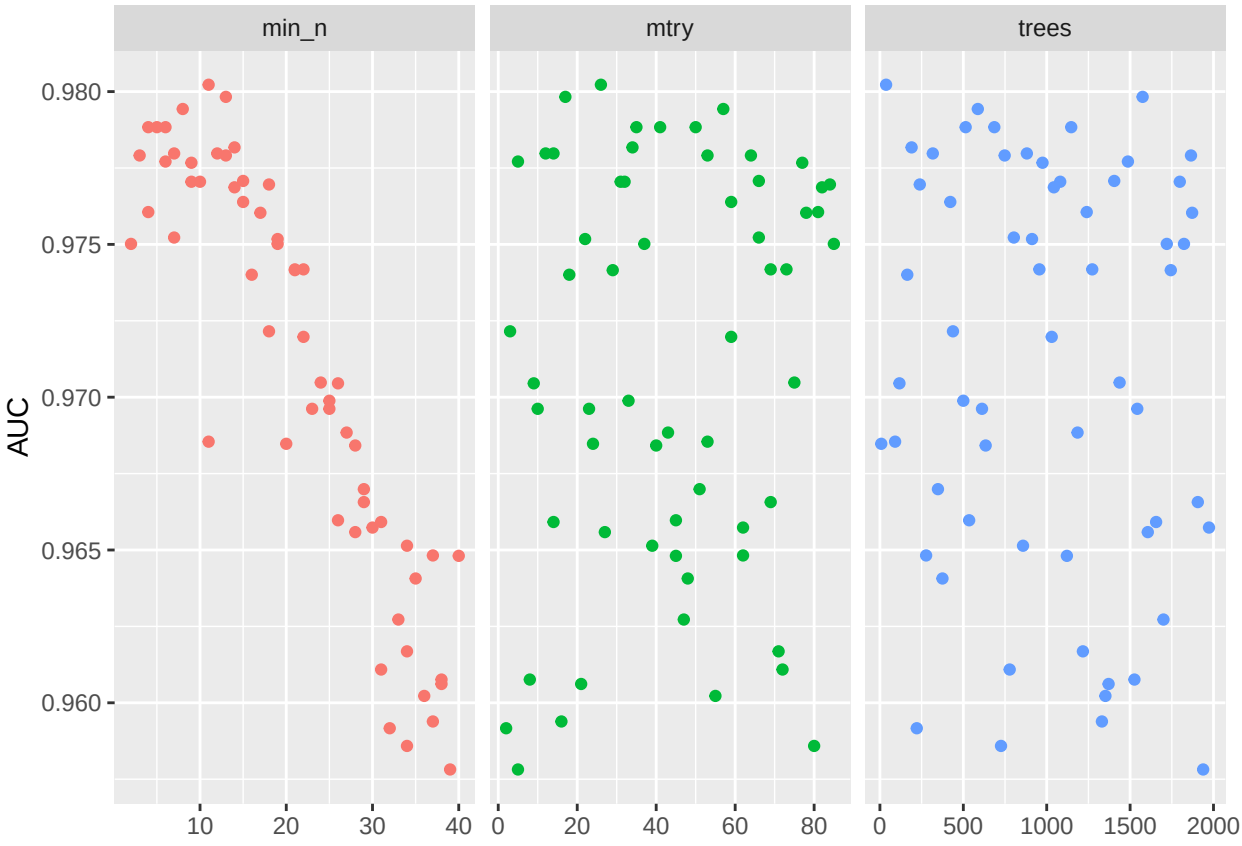
```
## i Creating pre-processing data to finalize unknown parameter: mtry
```

```
parallel::stopCluster(cl = my.cluster)
```

Optimal Hyperparameters

We can now see how the different models performed, we want to identify the variables that will lead to the highest roc_auc score.

```
# Visually determine the trend in the data to identify mtry and min_n to focus on
tune_res %>%
  collect_metrics() %>%
  filter(.metric == "roc_auc") %>%
  pivot_longer(min_n:mtry,
               values_to = "value",
               names_to = "parameter") %>%
  ggplot(aes(value, mean, color = parameter)) +
  geom_point(show.legend = FALSE) +
  facet_wrap(~parameter, scales = "free_x") +
  labs(x = NULL, y = "AUC")
```




```
best_auc <- select_best(tune_res, "roc_auc")
```

Fit the model

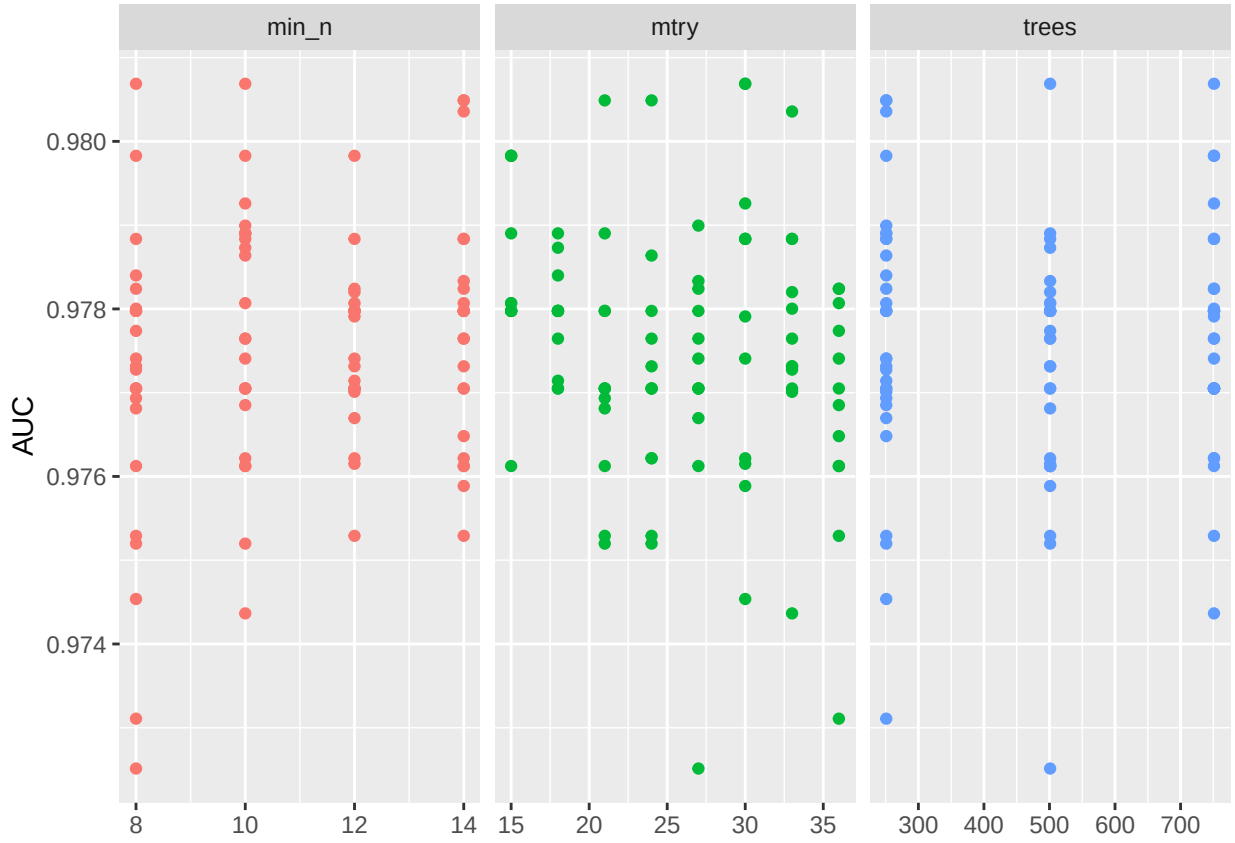
The best range of values for the grid are manually imputed and will be tested in a grid, this process could take several iterations as you fine tune the ideal hyperparmeters.

```
# Create grid that is more centered around the best hyperparameters
rf_grid <- expand.grid(min_n = seq(8, 14, 2),
                       mtry = seq(15, 36, 3),
                       trees = seq(251, 751, 250))

#create and register cluster
my.cluster <- parallel::makeCluster(n.cores)
doParallel::registerDoParallel(cl = my.cluster)

# Run the model again with the new grid
set.seed(24)
regular_res <- tune_grid(tune_wf,
                         resamples = trees_folds,
                         grid = rf_grid)

parallel::stopCluster(cl = my.cluster)
```

Interpret results

We will continue this process until it becomes "clear" that there are a set of hyperparmeters that perform better than any other combination and those values will be carried over to the final model.

```
# Visually determine the trend in the data to identify mtry and min_n to focus on
regular_res %>%
  collect_metrics() %>%
  filter(.metric == "roc_auc") %>%
  pivot_longer(min_n:mtry,
               values_to = "value",
               names_to = "parameter") %>%
  ggplot(aes(value, mean, color = parameter)) +
  geom_point(show.legend = FALSE) +
  facet_wrap(~parameter, scales = "free_x") +
  labs(x = NULL, y = "AUC")
```



Run the final model

Using the best hyperparmeters run the final model

```
best_auc <- select_best(regular_res, "roc_auc")

final_rf <- finalize_model(tune_spec, best_auc)
```

```
# Set up final workflow
final_wf <- workflow() %>%
  add_recipe(preprocessing_recipe) %>%
  add_model(final_rf)

final_res <- final_wf %>%
  last_fit(split)
```

model prediction

using the best random forest hyperparameters a new model is created, is is then used with the Test data from earlier to predict the tree classification from the unkown data set.

Using the confussion matrix we can see how well the model performed on predicting the tree type based on the data provided to it.

```
final_res %>%
  collect_metrics()
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy multiclass     0.982 Preprocessor1_Model1
## 2 roc_auc  hand_till      0.995 Preprocessor1_Model1
```

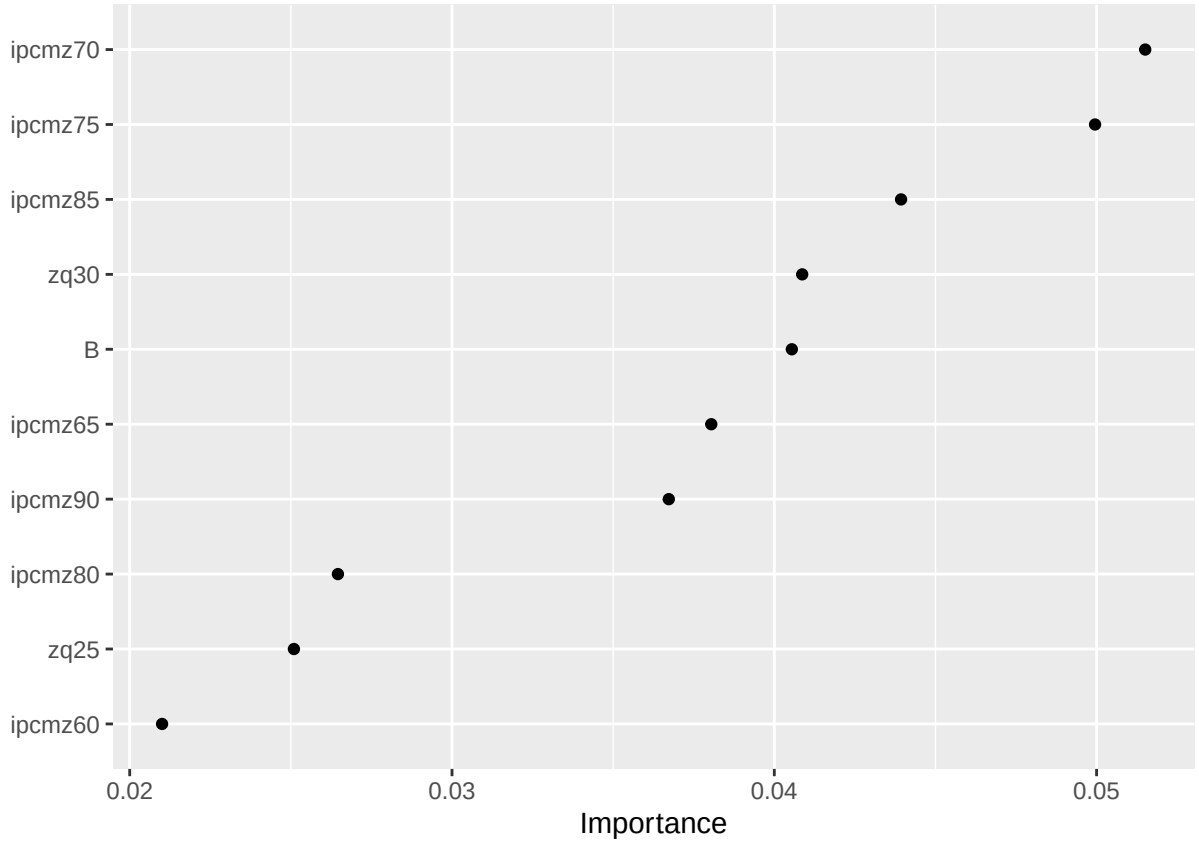
```
final_res %>%
  collect_predictions() %>%
  conf_mat(Tree_Type, .pred_class)
```

```
##            Truth
## Prediction  Deciduous Hemlock Pine
##   Deciduous        30       0    0
##   Hemlock           0      13    0
##   Pine              1       0   12
```

Variable Importance

Now that the models are complted we can plot how much each variable contributed to the overal performance of the model and see what the top contributers were.

```
final_rf %>%
  set_engine("ranger", importance = "permutation") %>%
  fit(Tree_Type ~ ., data = juice(tree_prep)) %>%
  vip(geom = "point")
```


ipcmz70
ipcmz75
ipcmz85
zq30
B
ipcmz65
ipcmz90
ipcmz80
zq25
ipcmz60
0.02
0.03
0.04
0.05
Importance