Machine Learning Methods for Predicting Forest Composition: A Final Project for Machine Learning Concepts and Analysis

Madeleine Desrochers

11/2/2021

Contents

Abstract

Increasing CO_2 concentrations in the atmosphere are causing changes in regional climate patterns. Changes in the temperature and moisture regimes of an area have an affect on the forest composition. Being able to track and predict these changes would allow resource managers and other forest stakeholders to prepare for the consequences of these changes. To test the feasibility of using climactic variables to predict the forest composition, Forest Inventory and Analysis data was combined with 13 climate variable and implemented in three machine learning algorithms - a decision tree, a random forest, and a stochastic gradient boosting machine. We found that the random forest model was the most effective at predicting forest composition, however all of these models could be implemented by forest managers as a strategy to predict and mitigate the effects of global climate change.

1. Introduction

In the last century, the world has seen the dramatic effects of a rapidly changing climate. Increased CO_2 concentrations are causing changes in climatic patterns and shifting regional temperature and precipitation regimes. As these changes become more prevalent, some species populations may see a decline in abundance and success, while others still become better competitors, or even may enlarge their range to colonize areas previously unavailable to them (Hansen et al. 2001). Changes in the composition and biodiversity of forests is likely to have effects across all levels of organization – individual, species, population, community and ecosystem scales.

The northern hardwood forest is most typically characterized by the maple/beech/ birch cover type. This cover type is under threat from a variety of factors, including both forest pathogens and anthroprogenic climate change. Being able to predict the composition of forest stands using climatic data would allow for forecasting of the expected changes in the biodiversity of the northern forest region. To accomplish this goal, Forest Inventory Analysis (FIA) data from Maine, New Hampshire, Vermont and New York, along with climate data from 1984 to 2014 (Bose et al. 2001) was used in three machine learning algorithms, a decision tree model, a random forest and a stochastic gradient boosting machine to see if the climatic variables could be used to predict the species composition of northeastern forest stands.

2. Methods

2.2 Data

Relative abundance of the beech/birch/maple cover type from 1984 – 2014 derived from FIA data, as well as associated climate data was used from 4 northern states - Maine, Vermont, New Hampshire and New York (Bose et al. 2018). Climate data was sourced from Daymet Earth Data (https://daymet.ornl.gov/) and System for Automated Geoscientific Analyses raster layers (Böhner, McCloy & Strobl 2006). In total the data contains 78,661 entries of 14 variables. Descriptions of the included variables are available in Table 1.

Table 1. Description of variables included in the data set.

Variable Name	Variable Description
Relative abundance of beech/birch/maple cover type	Relative overstory basal area of American beech, sugar maple, red maple and birch species
Plot longitude	Longitude of study plot
Plot latitude	Latitude of study plot
Plot ecoregion	Name of ecoregion
Growing degree days	Sum of the number of days with temperatures > 5º C in the previous 3 years
Frost free days	Sum of the number of days with temperatures > 0º C in the previous 3 years
Mean annual temperature	Mean annual temperature of the three previous years
Mean annual precipitation	Mean annual precipitation of the three previous years
Vapor pressure	Mean annual vapor pressure of the three previous years
Temperature deviation	Deviation from long term mean annual temperature to the subject year temperature
Precipitation deviation	Deviation from long term mean precipitation temperature to the subject year precipitation
Elevation	Elevation of sample plot
Terrain Wetness	Terrain wetness index value for study plot
Aspect	Slope aspect of study plot

2.3 Models

All three models evaluated were trained on a subset of the data set (80%). The remaining 20% of the data was used as a holdout set.

The first model was a simple decision tree model. The model was fit using the rpart package (Therneau et al. 2019).

The second model was a random forest, implemented with the ranger package (Wright et al.). The model was tuned using a grid search method with 90 combinations of 5 hyperparameters. The combination of hyperparameters that minimized RMSE was selected, and the final random forest was executed with 800 trees, with 13 variables being considered at each split, a minimum of 8 observations in each leaf node, and no replacement of samples in the training data set.

The final model was a stochastic gradient boosting machine implemented with the lightgbm package (Ke et al. 2021). Model tuning was completed using an iterative grid search that selected the parameters that produced the lowest RMSE values. The final model was fit to 2,000 trees allowed to grow to a depth of 63, with a learning rate of 0., and a minimum of 8 observations per leaf node. Each tree was fit to a bootstrap sample of 50% of the data with 70% of the predictors available to be used in each tree.

Each of the three models were evaluated using RMSE and MAE. Data preparation was completed with the tidyverse package (Wickham et al. 2021). All analysis was completed using the r statistical modeling software version 4.1.1 (R Core Team 2021).

3. Results

Model accuracy metrics for all models reported in Table 2. The random forest model had the lowest RMSE and MAE values of the three models. Both the GBM and the decision tree had substantially higher RMSE values than the random forest model, although the decision tree did have slightly lower RMSE and MAE values than the GBM.

Table 2. Accuracy metrics for decision tree, random forest and GBM models for predicting the retaliative abundance of the beech/birch/maple cover type based on climactic data.

Metric	Decision Tree	Random Forest	Light GBM
RMSE	34.578	16.922	32.972
MAE	30.235	13.064	26.915

4. Discussion Of the three methods implemented in this analysis, the random forest substantially outperformed the other two models. The GBM had slightly lower RMSE and MAE values than the decision tree, however it was still considerably less accurate than the random forest. The decision tree, unsurprisingly was not an effective predictor of forest composition.

For future implementation, an ensemble of these models could potentially improve the overall prediction accuracy. It is also likely that with additional tuning, the GBM model could be significantly improved. For this application, the benefits of tuning did not outweigh the required effort inputs.

This analysis is a proof of concept that climactic data could be used to predict forest changes in forest composition due to global climate change. The data set utilized here included the relative abundance of only one forest cover type, but a similar approach could be conducted with more varied cover type data. Predicting changes in forest composition could be applied in many sectors. Industries that rely on forest products need to be able to predict and plan for inevitable changes in the supply of their target species. Additionally, this method could also be applied to general forest management practices and be used to decrease the time and effort required for forest inventories.

5. Citations

Böhner, J., McCloy, K.R. & Strobl, J. (2006) SAGA – analysis and modelling applications. Göttinger Geographische Abhandlungen, 115, 130.

Bose, Arun K.; Weiskittel, Aaron; Wagner, Robert G. (2018), Data from: A three decade assessment of climate-associated changes in forest composition across the north-eastern USA, Dryad, Dataset, https://doi.org/10.5061/dryad.qj8gh

Hansen, A. J.; Neilson, R. P.; Dale, V. H.; Flather, C. H.; Iverson, L. R.; Currie, D. J.; Shafer, S.; Cook, R.; & Bartlein, P. J. (2001). Global Change in Forests: Responses of Species, Communities, and Biomes: Interactions between climate change and land use are projected to cause large shifts in biodiversity. BioScience, 51(9), 765–779. https://doi.org/10.1641/0006-3568(2001)051%5B0765:GCIFRO%5D2.0.CO;2

R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Therneau, Terry; Atkinson, Beth; Ripley, Brian. (2019). rpart: Recursive Partitioning and Regression Trees. https://cran.r-project.org/web/packages/rpart/index.html

Wickham et al., (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 1686, https://doi.org/10.21105/joss.01686

Wright, Marvin N.; Ziegler, Andreas (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. Journal of Statistical Software, 77(1), 1-17. doi:10.18637/jss.v077.i01

Yu Shi, Guolin Ke, Damien Soukhavong, James Lamb, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu and Nikita Titov (2021). lightgbm: Light Gradient Boosting Machine. R package version 3.3.0. https://CRAN.R-project.org/package=lightgbm

Appendix A: Code for ML Algorithms

```
library(here)
library(tidyverse)
org_forest_comp <- read_csv(here("MD_MLCA_Final_Project/","Data_Dryad_ARUN.csv"), col_names = TRUE)
forest_comp <- org_forest_comp[,c(18,30:42)]
forest_comp <- forest_comp[!duplicated(forest_comp),]

set.seed(123)
row_idx <- sample(seq_len(nrow(forest_comp)), nrow(forest_comp))
training <- forest_comp[row_idx < nrow(forest_comp) * 0.8,]
testing <- forest_comp[row_idx >= nrow(forest_comp) * 0.8,]
tuningsample <- forest_comp[row_idx < nrow(forest_comp) * 0.01,]

#decision tree
library(rpart)
library(rpart.plot)

decision_tree <- rpart(rela_mapBB_BA ~., data = training)

rpart(rela_mapBB_BA ~., data = training)|>
  rpart.plot(type=4)

dt_train_RMSE <- sqrt(mean((predict(decision_tree, training)-training$rela_mapBB_BA)^2))
dt_testRMSE <- sqrt(mean((predict(decision_tree, testing)-testing$rela_mapBB_BA)^2))

dt_testMAE <- sum(abs(testing$rela_mapBB_BA - predict(decision_tree, testing)))/nrow(testing)

#random forest

library(ranger)

rf_k_fold_cv <- function(data, k, ...) {
  per_fold <- floor(nrow(data) / k)
  fold_order <- sample(seq_len(nrow(data)),
                       size = per_fold * k)
  fold_rows <- split(
    fold_order,
    rep(1:k, each = per_fold)
  )
  vapply(
    fold_rows,
    \(fold_idx) {
      fold_test <- data[fold_idx, ]
      fold_train <- data[-fold_idx, ]
      fold_rf <- ranger(rela_mapBB_BA ~ ., fold_train)
      rf_calc_rmse(fold_rf, fold_test)
    },
    numeric(1)
  ) |>
    mean()
}

set.seed(123)
```

```r
rf_tuning_grid <- expand.grid(
  mtry = c(5, 10, 11, 12, 13),
  min.node.size = c(8, 9, 10),
  replace = c(TRUE,FALSE),
  sample.fraction = c(0.6, 0.8, 1),
  rmse = NA
)

for (i in seq_len(nrow(rf_tuning_grid))) {
  rf_tuning_grid$rmse[i] <- rf_k_fold_cv(
    tuningsample,
    k = 5,
    mtry = rf_tuning_grid$mtry[i],
    min.node.size = rf_tuning_grid$min.node.size[i],
    replace = rf_tuning_grid$replace[i],
    sample.fraction = rf_tuning_grid$sample.fraction[i]
  )
}

head(rf_tuning_grid[order(rf_tuning_grid$rmse), ])

final_rf <- ranger(
  rela_mapBB_BA ~ .,
  testing,
  num.trees = 800,
  mtry = 13,
  min.node.size = 8,
  replace = FALSE,
  sample.fraction = 0.8
)

rfRMSE <- rf_calc_rmse(final_rf, testing)

rf_final_predictions <- predictions(predict(final_rf, testing))
rfMAE <- sum(abs(testing$rela_mapBB_BA - rf_final_predictions))/nrow(testing)

#gbm
library(lightgbm)

lgb_forest_comp <- forest_comp %>%
  mutate(dummy_value = 1) %>%
  pivot_wider(
    names_from = eco_region,
    names_prefix = "eco_region",
    values_from = dummy_value,
    values_fill = 0
  )

set.seed(123)
row_idx <- sample(seq_len(nrow(lgb_forest_comp)), nrow(lgb_forest_comp))
lgb_training <- lgb_forest_comp[row_idx < nrow(lgb_forest_comp) * 0.8,]
lgb_testing <- lgb_forest_comp[row_idx >= nrow(lgb_forest_comp) * 0.8,]
```

```r
xtrain <- as.matrix(lgb_training[setdiff(names(lgb_training), "rela_mapBB_BA")])
ytrain <- lgb_training[["rela_mapBB_BA"]]
xtest <- as.matrix(lgb_testing[setdiff(names(lgb_testing), "rela_mapBB_BA")])

first_lgb <- lightgbm(
  data = xtrain,
  label = ytrain,
  verbose = -1L,
  obj = "regression",
)

lgb_predictions <- predict(first_lgb, xtest)
lgb_RMSE <- sqrt(mean((lgb_predictions - testing$rela_mapBB_BA)^2))

lgb_calc_rmse <- function(model, data) {
  xtest <- as.matrix(data[setdiff(names(data), "rela_mapBB_BA")])
  lgb_predictions <- predict(model, xtest)
  sqrt(mean((lgb_predictions - data$rela_mapBB_BA)^2))
}

lgb_k_fold_cv <- function(data, k, nrounds = 10L, ...) {
  per_fold <- floor(nrow(data) / k)
  fold_order <- sample(seq_len(nrow(data)),
                       size = per_fold * k)
  fold_rows <- split(
    fold_order,
    rep(1:k, each = per_fold)
  )
  vapply(
    fold_rows,
    \(fold_idx) {
      fold_test <- data[fold_idx, ]
      fold_train <- data[-fold_idx, ]
      xtrain <- as.matrix(fold_train[setdiff(names(fold_train),
                                             "rela_mapBB_BA")])
      ytrain <- fold_train[["rela_mapBB_BA"]]
      fold_lgb <- lightgbm(
        data = xtrain,
        label = ytrain,
        verbose = -1L,
        obj = "regression",
        nrounds = nrounds,
        params = ...
      )
      lgb_calc_rmse(fold_lgb, fold_test)
    },
    numeric(1)
  ) |>
    mean()
}

lgb_tuning_grid_1 <- expand.grid(
  learning_rate = 0.1,
  nrounds = c(10, 50, 100, 500, 1000, 1500, 2000, 2500, 3000),
  rmse = NA
)

for (i in seq_len(nrow(lgb_tuning_grid_1))) {
  lgb_tuning_grid_1$rmse[i] <- lgb_k_fold_cv(
    lgb_training,
    k = 5,
    learning_rate = lgb_tuning_grid_1$learning_rate[i],
    nrounds = lgb_tuning_grid_1$nrounds[i]
  )
}

head(arrange(lgb_tuning_grid_1, rmse), 2)

lgb_tuning_grid_2 <- expand.grid(
  learning_rate = 0.1,
  nrounds = 2000,
  max_depth = c(-1, 2, 8, 32, 63),
  min_data_in_bin = c(4, 6, 8, 10, 12),
  rmse = NA
)

for (i in seq_len(nrow(lgb_tuning_grid_2))) {
  lgb_tuning_grid_2$rmse[i] <- lgb_k_fold_cv(
    lgb_training,
    k = 5,
    learning_rate = lgb_tuning_grid_2$learning_rate[i],
    nrounds = lgb_tuning_grid_2$nrounds[i],
    max_depth = lgb_tuning_grid_2$max_depth[i],
    min_data_in_bin = lgb_tuning_grid_2$min_data_in_bin[i]
  )
}
head(arrange(lgb_tuning_grid_2, rmse), 2)

lgb_tuning_grid_3 <- expand.grid(
  learning_rate = 0.1,
  nrounds = 2000,
  max_depth = 63,
  min_data_in_bin = 8,
  bagging_freq = c(0, 1, 5, 10),
  bagging_fraction = seq(0.3, 1.0, 0.1),
  feature_fraction = seq(0.3, 1.0, 0.1),
  rmse = NA
)

for (i in seq_len(nrow(lgb_tuning_grid_3))) {
  lgb_tuning_grid_3$rmse[i] <- lgb_k_fold_cv(
    lgb_training,
    k = 5,
    learning_rate = lgb_tuning_grid_3$learning_rate[i],
    nrounds = lgb_tuning_grid_3$nrounds[i],
    max_depth = lgb_tuning_grid_3$max_depth[i],
    min_data_in_bin = lgb_tuning_grid_3$min_data_in_bin[i],
    bagging_freq = lgb_tuning_grid_3$bagging_freq[i],
    bagging_fraction = lgb_tuning_grid_3$bagging_fraction[i],
    feature_fraction = lgb_tuning_grid_3$feature_fraction[i]
  )
}
head(arrange(lgb_tuning_grid_3, rmse), 2) |>
  select(bagging_freq, bagging_fraction, feature_fraction, rmse)

final_lgb <- lightgbm(
  data = xtrain,
  label = ytrain,
  verbose = -1L,
  obj = "regression",
  nrounds = 2000,
  params = list(
    learning_rate = 0.1,
    max_depth = 63,
    min_data_in_bin = 8,
    bagging_freq = 0,
    bagging_fraction = 0.5,
    feature_fraction = 0.7
  )
)


lgbRMSE <- lgb_calc_rmse(final_lgb, lgb_testing)
lgb_predictions <- predict(final_lgb, xtest)
lgbMAE <- sum(abs(lgb_testing$rela_mapBB_BA - lgb_predictions))/nrow(lgb_testing)
```