Overview

The goal of rfvip is to tune and select a good Random Forest model based on the accuracy and variable importance metrics associated with each model. To accomplish this, functions are available to tabulate and plot results designed to help the user select an optimal model.

This package contains functions for assessing variable relations and associations prior to modeling with a Random Forest algorithm (although these are relevant for any predictive model). Metrics such as partial correlations and variance inflation factors are tabulated as well as plotted for the user using the functions partial\_cor() and robust\_vifs().

The function mtry\_compare() is available for tuning the hyper-parameter mtry based on model performance and variable importance metrics. This grid-search technique provides tables and plots showing the effect of mtry on each of the assessment metrics. It also returns each of the evaluated models to the user.

The package also provides superior ggplot2 variable importance plots for individual models using the function ggvip(). This function is a highly aesthetic and editable improvement upon the function randomForest::varImpPlot() and other basic importance graphics.

All of the plots generated by these functions are developed with ggplot2 techniques so that the user has the ability to edit and improve further upon the plots.

Example: Boston

This is a basic example which shows you how to solve a common problem:

**library**(rfvip)

**library**(MASS)

**library**(EZtune)

We will attempt to build an optimal model for the Boston housing data. This can be found in the MASS package. To begin we will run some preliminary diagnostics on our data.

set.seed(1234)

pcs <- partial\_cor(medv ~ ., data = Boston, model = lm)

pcs$plot\_y\_part\_cors

A graph with black dots

Description automatically generated

rv <- robust\_vifs(medv ~ ., data = Boston, model = lm)

rv$plot\_nonlin\_vifs

A graph with black dots

Description automatically generated

These look quite good with regard to collinearity. The VIFs are all less than 10. The partial correlations with the response are a type of pseudo-importance assessing the importance each variable does not share with the others. Now we tune our model across four mtry values.

set.seed(1)

m <- mtry\_compare(medv ~ ., data = Boston, sqrt = TRUE,

mvec = c(1,4,9,13), num\_var = 7)

m$gg\_model\_errors

A graph with a line

Description automatically generated

m$model\_errors

*#> mtry mse*

*#> 1 1 18.96051*

*#> 2 4 10.11247*

*#> 3 9 10.13187*

*#> 4 13 10.36482*

According to the accuracy plot and table, our best choice is when mtry is 4. However, the accuracy for the best model is very similar to two of the other models. We now look at the variable importance metrics across the different models.

m$gg\_var\_imp\_error

A graph of different colored lines

Description automatically generated

The top two variables are consistently identified as more important than the other variables and their order remains unchanged across mtry. However, the variables ‘nox’ and ‘dis’ switch order as mtry increases. Common sense suggests that pollution (nox) is correlated with distance to employment centers (dis). It can be assumed that most home buyers consider location to work more than pollution when selecting a house. Therefore, ‘dis’ is likely a more causal driver of price than ‘nox’. Consequently, the model where mtry is 9 appears to be superior to the model where mtry is 4 (even if it is slightly more accurate).

We now take our selected model and build individual importance plots for it.

g <- ggvip(m$rf9)$both\_vips

A graph of a graph with black dots

Description automatically generated

Looks great. We have used variable importance and accuracy metrics to pick a solid model for prediction and with reasonably useful importance values.

Example: Lichen

Here is another example. This time using classification data.

**library**(rfvip)

We will attempt to build an optimal model for the Lichen data. This can be found in the EZtune package. The response is presence or absence (coded 0 or 1) of a lichen species, Lobaria Oregana. To begin we will run preliminary diagnostics on our data.

set.seed(1234)

lichen <- EZtune::lichen[, -c(1, 3:8)]

pairs(lichen[, c(16, 20, 26)])

A graph of a graph with Great Pyramid of Giza in the background

Description automatically generated

cor(lichen[, c(16, 20, 26)])

*#> MinTempAve AmbVapPressAve Elevation*

*#> MinTempAve 1.0000000 0.9973158 -0.9781953*

*#> AmbVapPressAve 0.9973158 1.0000000 -0.9745659*

*#> Elevation -0.9781953 -0.9745659 1.0000000*

pcs <- partial\_cor(factor(LobaOreg) ~ ., data = lichen, model = lm,

num\_var = 15)

pcs$plot\_y\_part\_cors

A graph with black dots

Description automatically generated

rv <- robust\_vifs(factor(LobaOreg) ~ ., data = lichen, model = lm,

num\_var = 15)

rv$plot\_nonlin\_vifs

A graph with black dots and white text

Description automatically generated

These variables have large issues of collinearity. To illustrate this look at the pairs plots above for ‘MinTempAve’, ‘Elevation’, and ‘AmbVapPressAve’. Most of the VIFs exceed the standard threshold. The partial correlations with the response are a type of pseudo-importance assessing the importance each variable does not share with the others. Now we tune our model across four mtry values.

set.seed(100)

m <- mtry\_compare(factor(LobaOreg) ~ ., data = lichen, sqrt = TRUE,

mvec = c(1,5,19,33), num\_var = 7)

m$gg\_model\_errors

A graph with a line

Description automatically generated

m$model\_errors

*#> mtry misclass\_rate*

*#> 1 1 0.1797619*

*#> 2 5 0.1607143*

*#> 3 19 0.1571429*

*#> 4 33 0.1619048*

According to the accuracy plot and table, our best choice is when mtry is 19. However, the accuracy for the best model is very similar to some of the other models. We now look at the variable importance metrics across the different models.

m$gg\_var\_imp\_error

A graph of different colored lines

Description automatically generated

There are 3 variables to focus on. ‘MinTempAve’, ‘Elevation’, and ‘AmbVapPressAve’ were all shown to be highly correlated above. These variables appear to be the most importance variable when mtry is small. However, the importance of ‘Elevation’ drops off, and the importance of ‘AmbVapPressAve’ drops even more as mtry increases. After seeing these changes, a researcher might consider how these variable actually affect lichen. They would find that ‘MinTempAve’ informs freezing which directly causes lichen presence. They would also realize that ‘Elevation’ indirectly causes lichen presence since ‘Elevation’ drives ‘MinTempAve’. ‘AmbVapPressAve’ can be assumed to be a byproduct of ‘Elevation’ and is not a feature that should cause lichen presences. While it is highly predictive, it is not something a scientist would prescribe for inducing the response. As mtry increases, casual variables rise while collinear variables fall.

No solution is perfect, but mtry = 33 yields results that match the intuition about the effect our predictors have on the response.

We now take our selected model and build individual importance plots for it.

g <- ggvip(m$rf33)$both\_vips

A graph with black dots

Description automatically generated

*#g <- ggvip(m$rf33, num\_var = 13)$both\_vips*

Looks fantastic. We have used variable importance and accuracy metrics to pick a solid model for prediction and with much more useful importance values.