About the caret Package

Max Kuhn

max.kuhn@pfizer.com

Pfizer Global R&D
Research Statistics
Groton, CT

The Package

`caret` is short for **c**lassification **a**nd **re**gression **t**raining

It is not on CRAN yet, but it will be this year

It is a package full of miscellaneous functions that I find useful for building predictive models.

There is way more information and details in the package vignette. Load the package via `library(caret)` and type `vignette("caret")` to see it.

Pre–Processing

There are a few simple functions to pre–process data, such as centering and scaling

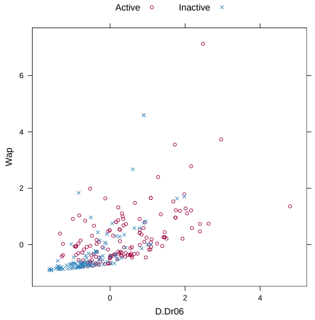
Also, there are some methods to do unsupervised feature selection:

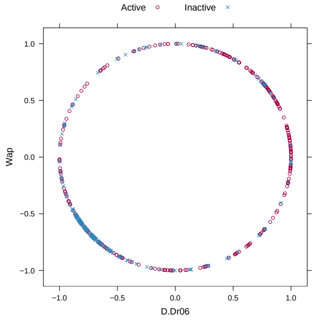
- If there are highly correlated predictors, as is the case in quantitative structure–activity relationship (QSAR) models and in gene expression studies, `caret` has an algorithm to identify a subset of predictors with absolute correlations below a threshold.
- There is a function to enumerate linear dependencies in predictors so that they can be removed.
- Also, there are cases where numeric predictors have sparse, discrete distributions. We call these “near–zero–variance” predictors. There is also a function to identify these.

Transforming Predictors

Transforming variables can help some models. One way to doing this the "spatial–sign" transformation. Let $\mathbf{x}$ be a vector containing the predictors for a single sample.

The transformation is $\mathbf{x}^* = \mathbf{x}/||\mathbf{x}||$. Samples are projected onto a unit circle:





Training Models

The main function in the package is called `train`. It has two main purposes:

1. To be a uniform interface to numerous regression and classification models. Many different models can be evaluated with minimal code modifications
2. To choose values for model tuning parameters (if any) using resampling techniques, such as cross–validation or bootstrapping.

There are similar R and Bioconductor packages: `ipred`, `e1071` and `MLInterfaces`.

Also, `caret` was built so that there is minimal effort to extending it to your favorite parallel processing library (such as `nws` or `Rlsf`)

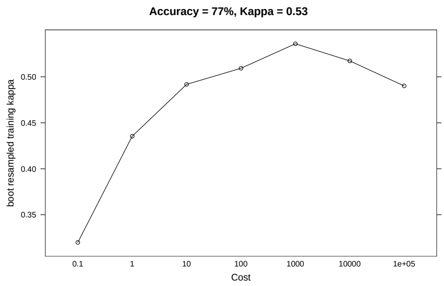
The Basic Idea

```
Create multiple splits or resamples of the data;
Create a grid of model complexity parameters;
for Each complexity parameter combination do
    for Each Data Split/Resample do
        Train a model with the current complexity parameter combination;
        Predict the held–back samples;
    end
    Calculate performance (e.g. accuracy, $R^2$) over the held–back samples;
end
Determine the complexity parameters with the best performance;
Refit the model using these parameters on the entire data set;
```

Example

As a QSAR example, the multidrug resistance reversal (MDRR) agent data was used to predict a specific type of chemical activity. Given a set of compounds with know activity data, the molecular structures were used to predict activity in new (or virtual) compounds.

To fit a support vector machine with a radial basis function, we need to determine the value of the cost (aka regularization) parameter. (There is a RBF parameter, but we fix that value up–front based on an analytical solution). We used bootstrapping:



Other Functions

There are a few different functions for data splitting (and a few more to come), a class for confusion matrices and functions to calculate ROC curves.

A wrapper for partial least squares is included so that there is a formula interface. This function also enables classification models using PLS.

There is a variable importance class. This has specific methods for several models (trees, bagged trees, boosted trees, random forests, MARS, PLS, OLS) and generic methods for other models.

There is a set of functions to apply RMA–like signal processing methods to Affymetrix gene chip data. This method is not batch–oriented, but does require a training set.