The caret Package: A Unified Interface for Predictive Models

Max Kuhn

Pfizer Global R&D
Nonclinical Statistics
Groton, CT
max.kuhn@pfizer.com

Motivation

Theorem (No Free Lunch)

In the absence of any knowledge about the prediction problem, no model can be said to be uniformly better than any other

Given this, it makes sense to use a variety of different models to find one that best fits the data

R has many packages for predictive modeling (aka machine learning)(aka pattern recognition) ...

Model Function Consistency

Since there are many modeling packages written by different people, there are some inconsistencies in how models are specified and predictions are made.

For example, many models have only one method of specifying the model (e.g. formula method only)

The table below shows the syntax to get probability estimates from several classification models:

Function	Syntax
`lda`	`predict.lda(...)` (no options needed)
`glm`	`predict.glm(..., type = "response")`
`gbm`	`predict.gbm(..., type = "response", n.trees)`
`mda`	`predict.mda(..., type = "posterior")`
`nnet`	`predict.nnet(..., type = "probs")`

The **caret** Package

The **caret** package was developed to:

- create a unified interface for modeling and prediction
- streamline model tuning using resampling
- provide a variety of "helper" functions and classes for day–to–day model building tasks
- increase computational efficiency using parallel processing

First commits within Pfizer: 6/2005

First version on CRAN: 10/2007

Website: http://caret.r-forge.r-project.org

JSS Paper: www.jstatsoft.org/v28/i05/paper

4 package vignettes (82 pages total)

Example Data

Kazius et al. (2005, *Journal of Medicinal Chemistry*) investigated models to use chemical structure to predict mutagenicity (the increase of mutations due to the damage to genetic material) as measured using an Ames test.

There were 4,337 compounds included in the data set with a mutagenicity rate of 55.3%.

Can we use chemical descriptors (e.g. molecular weight, number of hydrogen atoms) to predict mutagenicity?

The 1,576 descriptor values are contained `descr` and the outcome data are in a factor vector called `mutagen` with levels `"mutagen"` and `"nonmutagen"`.

These data are available from the package website.

Data Splitting

`createDataPartition` conducts stratified random splits

```
> set.seed(1)
> inTrain <- createDataPartition(mutagen, p = 3/4, list = FALSE)
> str(inTrain)

 int [1:3252, 1] 2 6 7 10 12 13 14 18 19 20 ...

> trainDescr <- descr[inTrain, ]
> testDescr <- descr[-inTrain, ]
> trainClass <- mutagen[inTrain]
> testClass <- mutagen[-inTrain]

> prop.table(table(mutagen))

mutagen
   mutagen nonmutagen
 0.5536332  0.4463668

> prop.table(table(trainClass))

trainClass
   mutagen nonmutagen
 0.5535055  0.4464945
```

Other functions: `createFolds`, `createResamples`

Data Pre–Processing Methods

`preProcess` calculates values that can be used to apply to any data set (e.g. training, set, unknowns).

Current methods: centering, scaling, spatial sign transformation, PCA or ICA "signal extraction"

```
> procValues <- preProcess(trainDescr, method = c("center", "scale"))
> procValues

Call:
preProcess.default(x = trainDescr, method = c("center", "scale"))

Created from 3252 samples and 1576 variables

> trainDescr <- predict(procValues, trainDescr)
> testDescr <- predict(procValues, testDescr)
```

Model Tuning

`train` uses resampling to tune and/or evaluate candidate models.

```
> rbfSVM <- train(x = trainDescr, y = trainClass,
+                 method = "svmRadial",
+                 tuneLength = 5,
+                 trControl = trainControl(method = "boot",
+                                          number = 50),
+                 fit = FALSE)
```

```
Fitting: sigma=0.0009351694, C=0.1
Fitting: sigma=0.0009351694, C=1
Fitting: sigma=0.0009351694, C=10
Fitting: sigma=0.0009351694, C=100
Fitting: sigma=0.0009351694, C=1000
```

`train` uses as many "tricks" as possible to reduce the number of models fits (e.g. using sub–models). Here, it uses the **kernlab** function `sigest` to analytically estimate the RBF scale parameter.

Model Tuning

```
> rbfSVM
```

```
3252 samples
1576 predictors

summary of bootstrap (50 reps) sample sizes:
    3252, 3252, 3252, 3252, 3252, 3252, ...

boot resampled training results across tuning parameters:

  C     sigma     Accuracy  Kappa  Accuracy SD  Kappa SD  Selected
  0.1   0.000935  0.723     0.442  0.723        0.442
  1     0.000935  0.818     0.632  0.818        0.632     *
  10    0.000935  0.813     0.623  0.813        0.623
  100   0.000935  0.801     0.597  0.801        0.597
  1000  0.000935  0.801     0.598  0.801        0.598

Accuracy was used to select the optimal model using the largest value.

The final values used in the model were C = 1 and sigma = 0.000935.
```

```
> class(rbfSVM$finalModel)
```

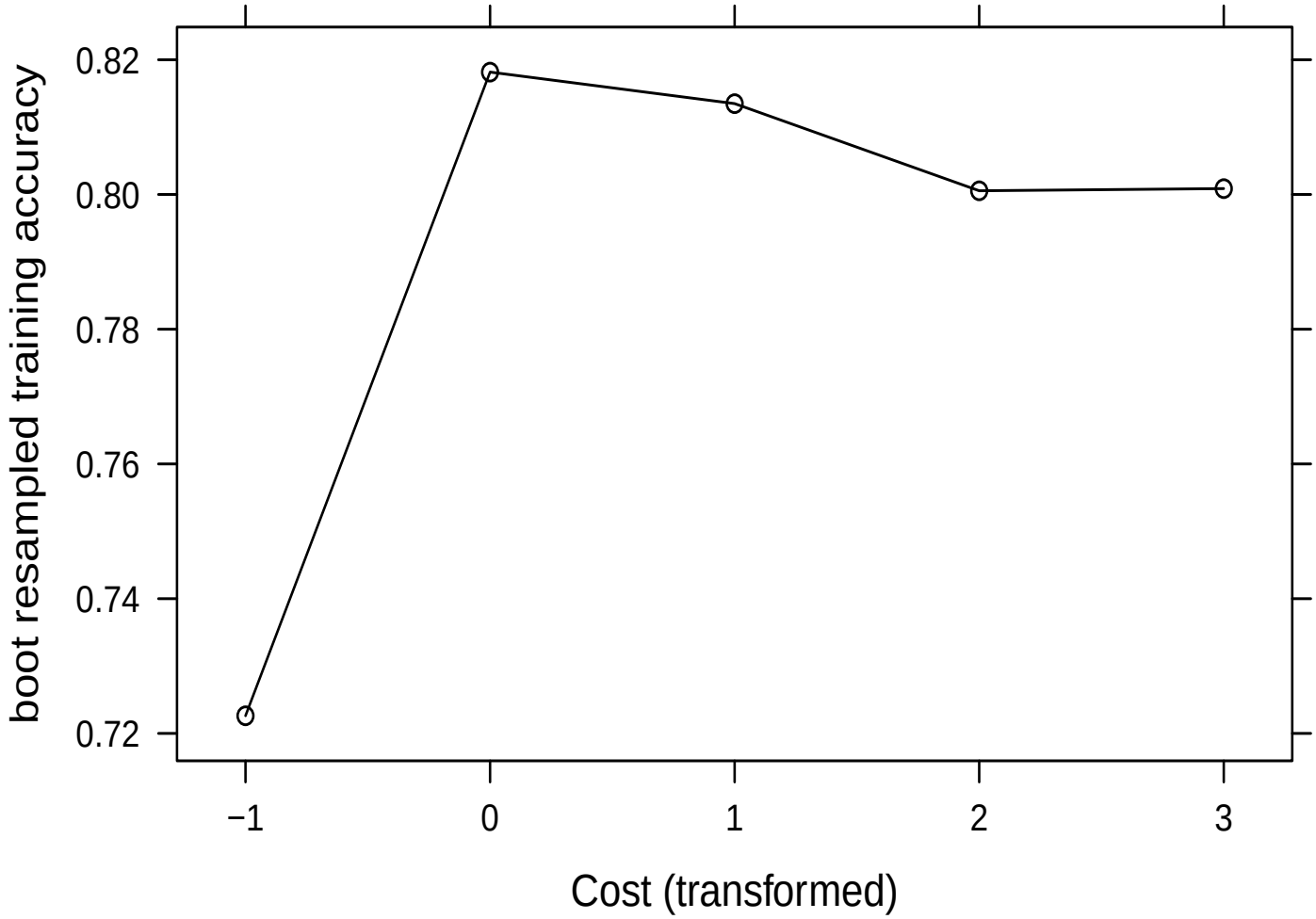
```
[1] "ksvm"
attr(,"package")
[1] "kernlab"
```

Model Tuning

- Currently, there are options for over 80 models (see `?train` for a list)
- Allows user–defined search grid, performance metrics and selection rules
- Easily integrates with any parallel processing framework that can emulate `lapply`
- Formula and non–formula interfaces
- Methods: `predict`, `print`, `plot`, `varImp`, `resamples`, `xyplot`, `densityplot`, `histogram`, `stripplot`, ...

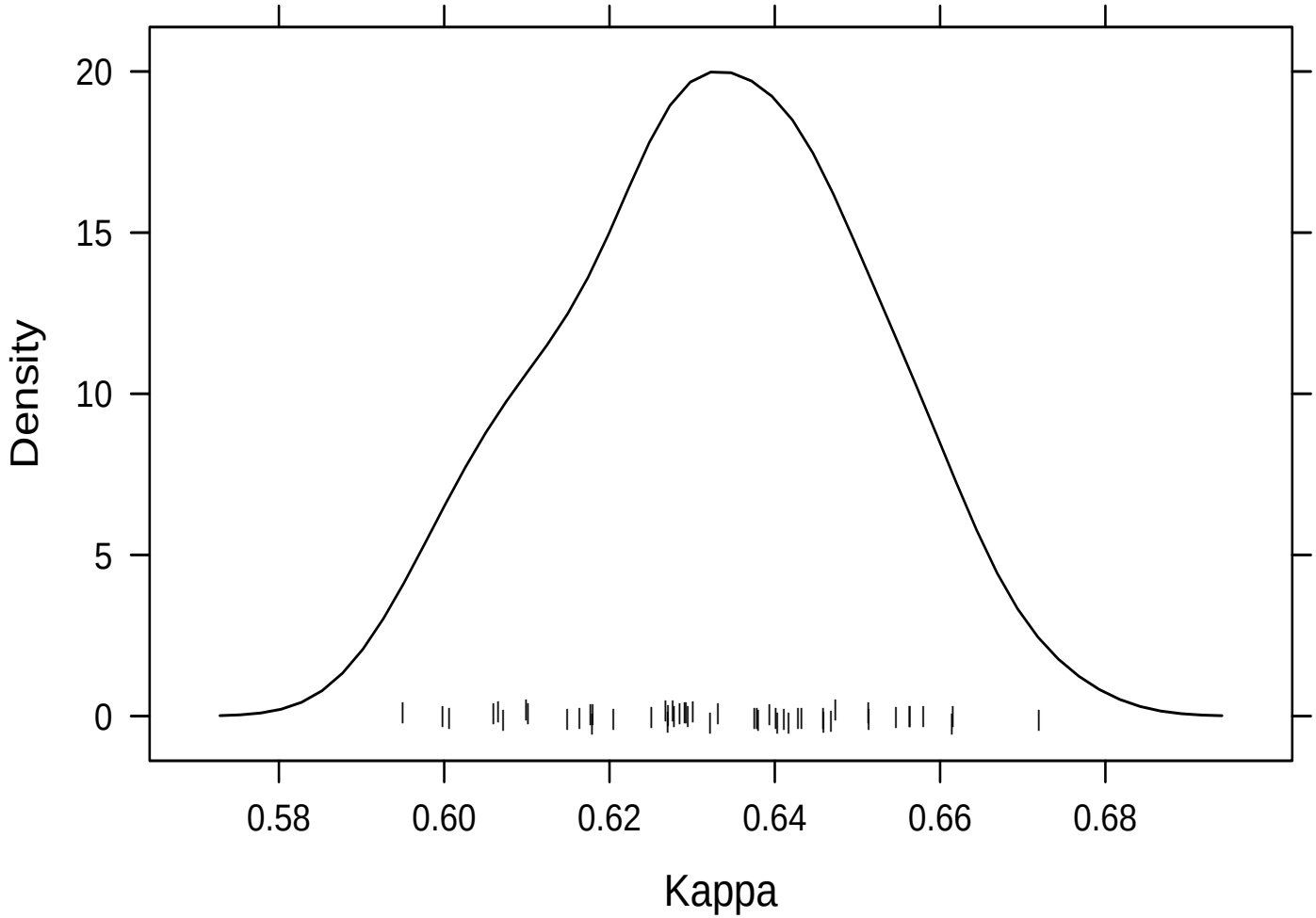
Plots

```
plot(rbfSVM, xTrans = function(x) log10(x))
```



Plots

```
densityplot(rbfSVM, metric = "Kappa", pch = "|")
```



Prediction and Performance Assessment

The `predict` method can be used to get results for other data sets:

```
> svmPred <- predict(rbfSVM, testDescr)
> str(svmPred)

 Factor w/ 2 levels "mutagen","nonmutagen": 1 2 2 2 1 1 2 2 2 2 ...

> svmProbs <- predict(rbfSVM, testDescr, type = "prob")
> str(svmProbs)

'data.frame': 1083 obs. of  2 variables:
 $ mutagen   : num  0.8139 0.4272 0.3121 0.0722 0.9678 ...
 $ nonmutagen: num  0.1861 0.5728 0.6879 0.9278 0.0322 ...
```

Predction and Performance Assessment

```
> confusionMatrix(svmPred, testClass)
Confusion Matrix and Statistics

            Reference
Prediction   mutagen nonmutagen
  mutagen        519          92
  nonmutagen      81         391

               Accuracy : 0.8403
                 95% CI : (0.8171, 0.8616)
    No Information Rate : 0.554
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.676

            Sensitivity : 0.8650
            Specificity : 0.8095
         Pos Pred Value : 0.8494
         Neg Pred Value : 0.8284
             Prevalence : 0.5540
         Detection Rate : 0.4792
   Detection Prevalence : 0.5642

       'Positive' Class : mutagen
```

Other Functions and Classes

- `nearZeroVar`: a function to remove predictors that are sparse and highly unbalanced
- `findCorrelation`: a function to remove the optimal set of predictors to achieve low pair–wise correlations
- `predictors`: class for determining which predictors are included in the prediction equations (e.g. `rpart`, `earth`, `lars` models) (currently 52 methods)
- `resamples`: a class for visualizing and assessing model comparisons using resampled values
- `confusionMatrix`, `sensitivity`, `specificity`, `posPredValue`, `negPredValue`: classes for assessing classifier performance
- `varImp`: classes for assessing the aggregate effect of a predictor on the model equations (currently 15 methods)

Other Functions and Classes

- `knnreg`: nearest–neighbor regression
- `plsda`, `splsda`: PLS discriminant analysis
- `icr`: independent component regression
- `pcaNNet`: `nnet:::nnet` with automatic PCA pre–processing step
- `bagEarth`, `bagFDA`: bagging with MARS and FDA models
- `normalize2Reference`: RMA–like processing of Affy arrays using a training set
- `spatialSign`: class for transforming numeric data ($x' = x/||x||$)
- `maxDissim`: a function for maximum dissimilarity sampling
- `rfe`: a class/framework for recursive feature selection (RFE) with external resampling step
- `sbf`: a class/framework for applying univariate filters prior to predictive modeling with external resampling
- `featurePlot`: a wrapper for several `lattice` functions

Thanks

useR! Organizers

R Core

Pfizer's Statistics leadership for providing the time and support to create R packages

caret contributors: Jed Wing, Steve Weston, Andre Williams, Chris Keefer and Allan Engelhardt