Package: A3 (via r-universe)

September 2, 2024

Type Package

Title Accurate, Adaptable, and Accessible Error Metrics for Predictive Models

Version 1.0.0

Date 2015-08-15

Author Scott Fortmann-Roe

Maintainer Scott Fortmann-Roe <scottfr@berkeley.edu>

Description Supplies tools for tabulating and analyzing the results of predictive models. The methods employed are applicable to virtually any predictive model and make comparisons between different methodologies straightforward.

License GPL (>= 2)

Depends R (>= 2.15.0), xtable, pbapply

Suggests randomForest, e1071

NeedsCompilation no

Date/Publication 2015-08-16 23:05:52

Repository https://scottfr.r-universe.dev

RemoteUrl https://github.com/cran/A3

RemoteRef HEAD

RemoteSha 26847d7d2053f3666604b369aa34e455d6c2d9e3

Contents

A3-package 2
a3 3
a3.base 4
a3.gen.default 5
a3.lm 7
a3.r2 8
housing 8
multifunctionality 9

plot.A3 10
plotPredictions 10
plotSlopes 11
print.A3 12
xtable.A3 12

Index **13**

`A3-package` *A3 Error Metrics for Predictive Models*

Description

A package for the generation of accurate, accessible, and adaptable error metrics for developing high quality predictions and inferences. The name A3 (pronounced "A-Cubed") comes from the combination of the first letters of these three primary adjectives.

Details

The overarching purpose of the outputs and tools in this package are to make the accurate assessment of model errors more accessible to a wider audience. Furthermore, a standardized set of reporting features are provided by this package which create consistent outputs for virtually any predictive model. This makes it straightforward to compare, for instance, a linear regression model to more exotic techniques such as Random forests or Support vector machines.

The standard outputs for each model fit provided by the A3 package include:

- Average Slope: Equivalent to a linear regression coefficient.
- Cross Validated R^2: Robust calculation of R^2 (percent of squared error explained by the model compared to the null model) values adjusting for over-fitting.
- p Values: Robust calculation of p-values requiring no parametric assumptions other than independence between observations (which may be violated if compensated for).

The primary functions that will be used are `a3` for arbitrary modeling functions and `a3.lm` for linear models. This package also includes `print.A3` and `plot.A3` for outputting the A3 results.

Author(s)

Scott Fortmann-Roe `<scottfr@berkeley.edu>` `http://Scott.Fortmann-Roe.com`

a3 *A3 Results for Arbitrary Model*

Description

This function calculates the A3 results for an arbitrary model construction algorithm (e.g. Linear Regressions, Support Vector Machines or Random Forests). For linear regression models, you may use the a3.lm convenience function.

Usage

```
a3(formula, data, model.fn, model.args = list(), ...)
```

Arguments

formula	the regression formula.
data	a data frame containing the data to be used in the model fit.
model.fn	the function to be used to build the model.
model.args	a list of arguments passed to model.fn.
...	additional arguments passed to a3.base.

Value

S3 A3 object; see a3.base for details

References

Scott Fortmann-Roe (2015). Consistent and Clear Reporting of Results from Diverse Modeling Techniques: The A3 Method. Journal of Statistical Software, 66(7), 1-23. <http://www.jstatsoft.org/v66/i07/>

Examples

```
## Standard linear regression results:

summary(lm(rating ~ ., attitude))

## A3 Results for a Linear Regression model:

# In practice, p.acc should be <= 0.01 in order
# to obtain finer grained p values.

a3(rating ~ ., attitude, lm, p.acc = 0.1)


## A3 Results for a Random Forest model:

# It is important to include the "+0" in the formula
# to eliminate the constant term.
```

```
require(randomForest)
a3(rating ~ .+0, attitude, randomForest, p.acc = 0.1)

# Set the ntrees argument of the randomForest function to 100

a3(rating ~ .+0, attitude, randomForest, p.acc = 0.1, model.args = list(ntree = 100))

# Speed up the calculation by doing 5-fold cross-validation.
# This is faster and more conservative (i.e. it should over-estimate error)

a3(rating ~ .+0, attitude, randomForest, n.folds = 5, p.acc = 0.1)

# Use Leave One Out Cross Validation. The least biased approach,
# but, for large data sets, potentially very slow.

a3(rating ~ .+0, attitude, randomForest, n.folds = 0, p.acc = 0.1)

## Use a Support Vector Machine algorithm.

# Just calculate the slopes and R^2 values, do not calculate p values.

require(e1071)
a3(rating ~ .+0, attitude, svm, p.acc = NULL)
```

a3.base *Base A3 Results Calculation*

Description

This function calculates the A3 results. Generally this function is not called directly. It is simpler to use a3 (for arbitrary models) or a3.lm (specifically for linear regressions).

Usage

```
a3.base(formula, data, model.fn, simulate.fn, n.folds = 10,
  data.generating.fn = replicate(ncol(x), a3.gen.default), p.acc = 0.01,
  features = TRUE, slope.sample = NULL, slope.displacement = 1)
```

Arguments

formula	the regression formula.
data	a data frame containing the data to be used in the model fit.
model.fn	function used to generate a model.
simulate.fn	function used to create the model and generate predictions.
n.folds	the number of folds used for cross-validation. Set to 0 to use Leave One Out Cross Validation.

`data.generating.fn`
: the function used to generate stochastic noise for calculation of exact p values.

`p.acc`
: the desired accuracy for the calculation of exact p values. The entire calculation process will be repeated $1/p.acc$ times so this can have a dramatic affect on time required. Set to `NULL` to disable the calculation of p values.

`features`
: whether to calculate the average slopes, added R^2 and p values for each of the features in addition to the overall model.

`slope.sample`
: if not NULL the sample size for use to calculate the average slopes (useful for very large data sets).

`slope.displacement`
: the amount of displacement to take in calculating the slopes. May be a single number in which case the same slope is applied to all features. May also be a named vector where there is a name for each feature.

Value

S3 `A3` object containing:

`model.R2`
: The cross validated R^2 for the entire model.

`feature.R2`
: The cross validated R^2's for the features (if calculated).

`model.p`
: The p value for the entire model (if calculated).

`feature.p`
: The p value for the features (if calculated).

`all.R2`
: The R^2's for the model features, and any stochastic simulations for calculating exact p values.

`observed`
: The observed response for each observation.

`predicted`
: The predicted response for each observation.

`slopes`
: Average slopes for each of the features (if calculated).

`all.slopes`
: Slopes for each of the observations for each of the features (if calculated).

`table`
: The A3 results table.

`a3.gen.default` *Stochastic Data Generators*

Description

The stochastic data generators generate stochastic noise with (if specified correctly) the same properties as the observed data. By replicating the stochastic properties of the original data, we are able to obtain the exact calculation of p values.

Usage

```
a3.gen.default(x, n.reps)
```

Arguments

`x` the original (observed) data series.

`n.reps` the number of stochastic repetitions to generate.

Details

Generally these will not be called directly but will instead be passed to the `data.generating.fn` argument of `a3.base`.

Value

A list of of length `n.reps` of vectors of stochastic noise. There are a number of different methods of generating noise:

`a3.gen.default` The default data generator. Uses `a3.gen.bootstrap`.

`a3.gen.resample` Reorders the original data series.

`a3.gen.bootstrap` Resamples the original data series with replacement.

`a3.gen.normal` Calculates the mean and standard deviation of the original series and generates a new series with that distribution.

`a3.gen.autocor` Assumesa first order autocorrelation of the original series and generates a new series with the same properties.

Examples

```
# Calculate the A3 results assuming an auto-correlated set of observations.
# In usage p.acc should be <=0.01 in order to obtain more accurate p values.

a3.lm(rating ~ ., attitude, p.acc = 0.1,
  data.generating.fn = replicate(ncol(attitude), a3.gen.autocor))


## A general illustration:

# Take x as a sample set of observations for a feature
x <- c(0.349, 1.845, 2.287, 1.921, 0.803, 0.855, 2.368, 3.023, 2.102, 4.648)

# Generate three stochastic data series with the same autocorrelation properties as x
rand.x <- a3.gen.autocor(x, 3)

plot(x, type=”l”)
for(i in 1:3) lines(rand.x[[i]], lwd = 0.2)
```

a3.lm *A3 for Linear Regressions*

Description

This convenience function calculates the A3 results specifically for linear regressions. It uses R's `glm` function and so supports logistic regressions and other link functions using the `family` argument. For other forms of models you may use the more general `a3` function.

Usage

```
a3.lm(formula, data, family = gaussian, ...)
```

Arguments

`formula`	the regression formula.
`data`	a data frame containing the data to be used in the model fit.
`family`	the regression family. Typically 'gaussian' for linear regressions.
`...`	additional arguments passed to `a3.base`.

Value

S3 `A3` object; see `a3.base` for details

Examples

```
## Standard linear regression results:

summary(lm(rating ~ ., attitude))

## A3 linear regression results:

# In practice, p.acc should be <= 0.01 in order
# to obtain fine grained p values.

a3.lm(rating ~ ., attitude, p.acc = 0.1)

# This is equivalent both to:

a3(rating ~ ., attitude, glm, model.args = list(family = gaussian), p.acc = 0.1)

# and also to:

a3(rating ~ ., attitude, lm, p.acc = 0.1)
```

a3.r2 *Cross-Validated Rˆ2*

Description

Applies cross validation to obtain the cross-validated R^2 for a model: the fraction of the squared error explained by the model compared to the null model (which is defined as the average response). A pseudo R^2 is implemented for classification.

Usage

```
a3.r2(y, x, simulate.fn, cv.folds)
```

Arguments

`y`	a vector or responses.
`x`	a matrix of features.
`simulate.fn`	a function object that creates a model and predicts y.
`cv.folds`	the cross-validation folds.

Value

A list comprising of the following elements:

`R2`	the cross-validated R^2
`predicted`	the predicted responses
`observed`	the observed responses

housing *Boston Housing Prices*

Description

A dataset containing the prices of houses in the Boston region and a number of features. The dataset and the following description is based on that provided by UCI Machine Learning Repository (`http://archive.ics.uci.edu/ml/datasets/Housing`).

Usage

```
data(housing)
```

Details

- CRIME: Per capita crime rate by town
- ZN: Proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS: Proportion of non-retail business acres per town
- CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
- NOX: Nitrogen oxides pollutant concentration (parts per 10 million)
- ROOMS: Average number of rooms per dwelling
- AGE: Proportion of owner-occupied units built prior to 1940
- DISTANCE: Weighted distances to five Boston employment centres
- HIGHWAY: Index of accessibility to radial highways
- TAX: Full-value property-tax rate per ten thousand dollar
- PUPIL.TEACHER: Pupil-teacher ratio by town
- MINORITY: 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
- LSTAT: Percent lower status of the population
- MED.VALUE: Median value of owner-occupied homes in thousands of dollars

References

Frank, A. & Asuncion, A. (2010). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Harrison, D. and Rubinfeld, D.L. Hedonic prices and the demand for clean air, J. Environ. Economics & Management, vol.5, 81-102, 1978.

`multifunctionality` *Ecosystem Multifunctionality*

Description

This dataset relates multifunctionality to a number of different biotic and abiotic features in a global survey of drylands. The dataset was obtained from (`http://www.sciencemag.org/content/335/6065/214/suppl/DC1`). The dataset contains the features listed below.

Usage

```
data(multifunctionality)
```

Details

- ELE: Elevation of the site
- LAT & LONG: Location of the site
- SLO: Site slope
- SAC: Soil sand content
- PCA_C1, PCA_C2, PCA_C3, PCA_C4: Principal components of a set of 21 climatic features
- SR: Species richness
- MUL: Multifunctionality

References

Maestre, F. T., Quero, J. L., Gotelli, N. J., Escudero, A., Ochoa, V., Delgado-Baquerizo, M., et al. (2012). Plant Species Richness and Ecosystem Multifunctionality in Global Drylands. Science, 335(6065), 214-218. doi:10.1126/science.1215442

`plot.A3` *Plot A3 Results*

Description

Plots an 'A3' object results. Displays predicted versus observed values for each observation along with the distribution of slopes measured for each feature.

Usage

```
## S3 method for class 'A3'
plot(x, ...)
```

Arguments

`x`	an A3 object.
`...`	additional options provided to `plotPredictions`, `plotSlopes` and `plot` functions.

Examples

```
 data(housing)
res <- a3.lm(MED.VALUE ~ NOX + ROOMS + AGE + HIGHWAY + PUPIL.TEACHER, housing, p.acc = NULL)
 plot(res)
```

`plotPredictions` *Plot Predicted versus Observed*

Description

Plots an 'A3' object's values showing the predicted versus observed values for each observation.

Usage

```
plotPredictions(x, show.equality = TRUE, xlab = "Observed Value",
  ylab = "Predicted Value", main = "Predicted vs Observed", ...)
```

Arguments

`x`	an A3 object,
`show.equality`	if true plot a line at 45-degrees.
`xlab`	the x-axis label.
`ylab`	the y-axis label.
`main`	the plot title.
`...`	additional options provided to the `plot` function.

Examples

```
data(multifunctionality)
 x <- a3.lm(MUL ~ ., multifunctionality, p.acc = NULL, features = FALSE)
 plotPredictions(x)
```

`plotSlopes` *Plot Distribution of Slopes*

Description

Plots an 'A3' object's distribution of slopes for each feature and observation. Uses Kernel Density Estimation to create an estimate of the distribution of slopes for a feature.

Usage

```
plotSlopes(x, ...)
```

Arguments

`x`	an A3 object.
`...`	additional options provided to the `plot` and `density` functions.

Examples

```
 require(randomForest)
 data(housing)

 x <- a3(MED.VALUE ~ NOX + PUPIL.TEACHER + ROOMS + AGE + HIGHWAY + 0,
   housing, randomForest, p.acc = NULL, n.folds = 2)

 plotSlopes(x)
```

print.A3 *Print Fit Results*

Description

Prints an ’A3’ object results table.

Usage

```
## S3 method for class 'A3'
print(x, ...)
```

Arguments

x	an A3 object.
...	additional arguments passed to the print function.

Examples

```
x <- a3.lm(rating ~ ., attitude, p.acc = NULL)
 print(x)
```

xtable.A3 *Nicely Formatted Fit Results*

Description

Creates a LaTeX table of results. Depends on the **xtable** package.

Usage

```
## S3 method for class 'A3'
xtable(x, ...)
```

Arguments

x	an A3 object.
...	additional arguments passed to the print.xtable function.

Examples

```
x <- a3.lm(rating ~ ., attitude, p.acc = NULL)
 xtable(x)
```

Index

∗ **datasets**
housing, 8
multifunctionality, 9

a3, *2*, 3, *4*, *7*
A3-package, 2
a3.base, *3*, 4, *6*, *7*
a3.gen.autocor *(*a3.gen.default*)*, 5
a3.gen.bootstrap *(*a3.gen.default*)*, 5
a3.gen.default, 5
a3.gen.normal *(*a3.gen.default*)*, 5
a3.gen.resample *(*a3.gen.default*)*, 5
a3.lm, *2–4*, 7
a3.r2, 8

density, *11*

glm, *7*

housing, 8

multifunctionality, 9

plot, *10*, *11*
plot.A3, *2*, 10
plotPredictions, *10*, 10
plotSlopes, *10*, 11
print, *12*
print.A3, *2*, 12
print.xtable, *12*

xtable.A3, 12