GARVaS: Genetic Algorithm for Regression Variable Selection

2014-12-12

Authors

- Yuan He
- Kuan-Cheng Lai
- Eugene Yedvabny

Code

The source code is available on GitHub at https://github.com/eyedvabny/GARVaS and is provided as a fully-functional R package. If the `devtools` package is available, GARVaS is easily installed by running:

```
devtools::install_github("eyedvabny/GARVaS")
```

or by cloning the repository and opening the included .Rproj file.The package depends on `doParallel`,`foreach` and `faraway` packages, which will be installed alongside GARVaS.

Introduction

GARVaS is an implementation of a Darwinian genetic algorithm for selecting best predictor variables for a linear or a generalized linear regression. Regression studies of multivariable datasets are often plagued by colinearity and confounding between the predictors, contributing to low predictive confidence.

A common approach to finding the best predictors in a large pool of candidates is an iterative process that starts with a full-interaction model and compares the quality of the fit after sequentially subtracting a predictor from the model. Unfortunately this process does not permute predictors, so good predictors can be eliminated without much improvement to the fitness if a bad predictor is still part of the model.

The genetic algorithm attemps to solve the above problem by keeping track of all the predictors and permuting them until the best combination emerges. Each predictor is a *gene* on a *chromosome* representing a model. A provided fitness function, e.g. AIC, determines the quality of that chromosome. At every generation the fittest chromosomes are stochastically crossed and mutated until "natural selection" converges on a *fittest individual*: a model with the best predictive power for the specified response.

Code Layout

We have auxiliary functions to carry out initialization, mutation, crossover, selection of parents and production. Parallel processing is used in both production and selection of parents. Generic fitness function and a possible choice of `glm()` are included as options for the selection function.

Our main function `select()` takes advantage of the above auxiliary functions and does variable selection of the given dataset using fitness function and linear model chosen by users. `select()` returns an S3 object with desired information stored as attributes. Printing and plotting methods are available for the S3 object.

Testing

Testing is done using the `testthat` package and consists of multiple unit tests for each of the underlying functions. The tests live in `test\testthat\test-###.R` files where ### corresponds to the function tested within that file. Due the stochastic nature of the algorithm the majority of the tests verify the consistency of the data structures and that *on average* mutations and cross-overs do not exceed the desired thresholds.

Testing of the main `select()` function is a combination of unit tests for validity of the underlying components as well as a "known truth" dataset that should converge to an expected model. An example of GARVaS usage is also illustrated in the following section.

All unit tests can be executed by running `devtools::test()` from the package directory.

Distribution of Work

Eugene did the R packages setup, initiated collaboration via Github, wrote the tests and improved the code. Yuan wrote the algorithm and carried out the implementation. Kuan-Cheng wrote the roxygen2 documentation and modified the algorithm. The three of us finished the paper documentation together.

Algorithm Implementation

We want to implement our algorithm on the built-in dataset within R called "mtcars", with the first column (mpg=miles per gallon) being the Y variable and the rest 10 columns being predictors.

First take a look at a portion of the dataset:

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

Then we compare the full model with the updated model using the genetic algorithm. Since the dataset is small, we set generations to be 30 to maximize convergence.

```r
mod <- lm(mpg~., data = mtcars) #Full model
result <- select(mtcars, generations = 30)
print(result)
```

```
## Fitness Score:
## [1] 154.1194
## Final Model:
## (Intercept)          wt        qsec          am
##    9.617781   -3.916504    1.225886    2.935837
```

Then we want to take a look at the variables chosen by the algorithm:

	Chr 1	Chr 2	Chr 3	Chr 4	Chr 5	Chr 6	Chr 7	Chr 8
cyl	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
disp	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
hp	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
drat	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
wt	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
qsec	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
vs	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
am	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
gear	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
carb	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Only 3 variables out of 10 were chosen, dimension got reduced significantly.

Now check the AIC and prediction accuracy of both models:

```
AIC(mod)
```

```
## [1] 163.7098
```

```
AIC(result$model)
```

```
## [1] 154.1194
```

```
MSE <- sum((predict(mod)-mtcars[,1])^2)/nrow(mtcars)
MSE
```

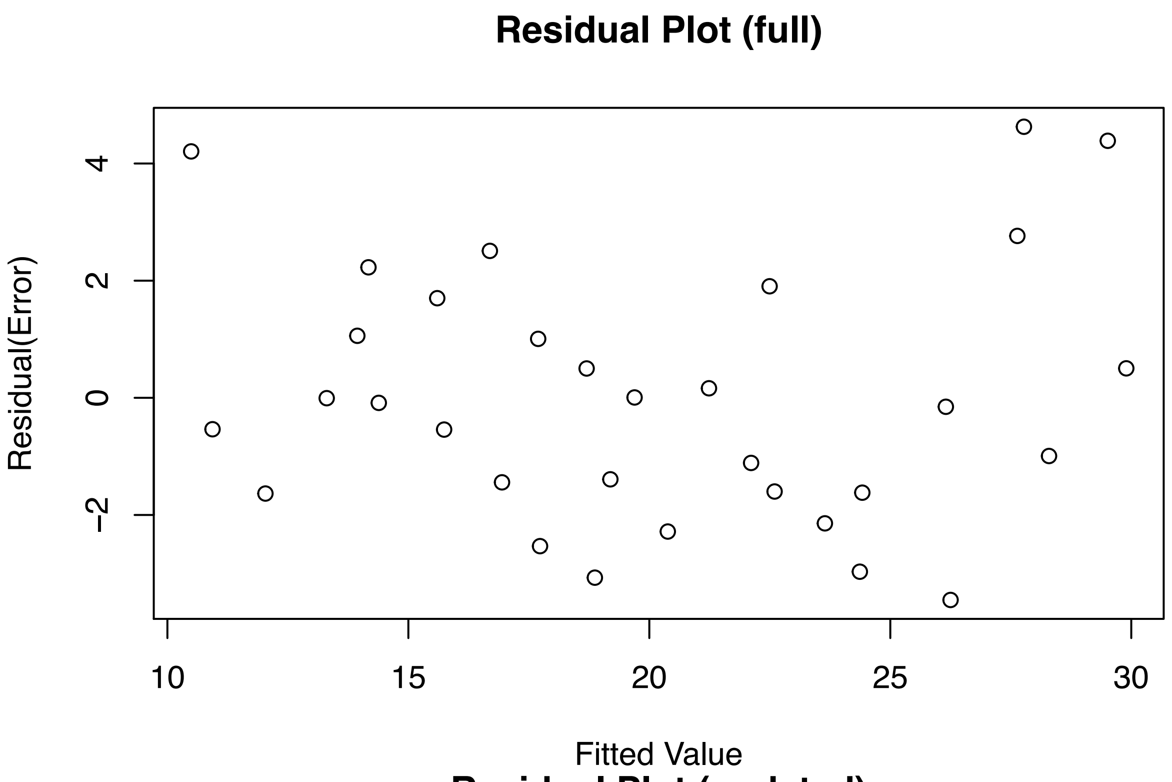
```
## [1] 4.609201
```

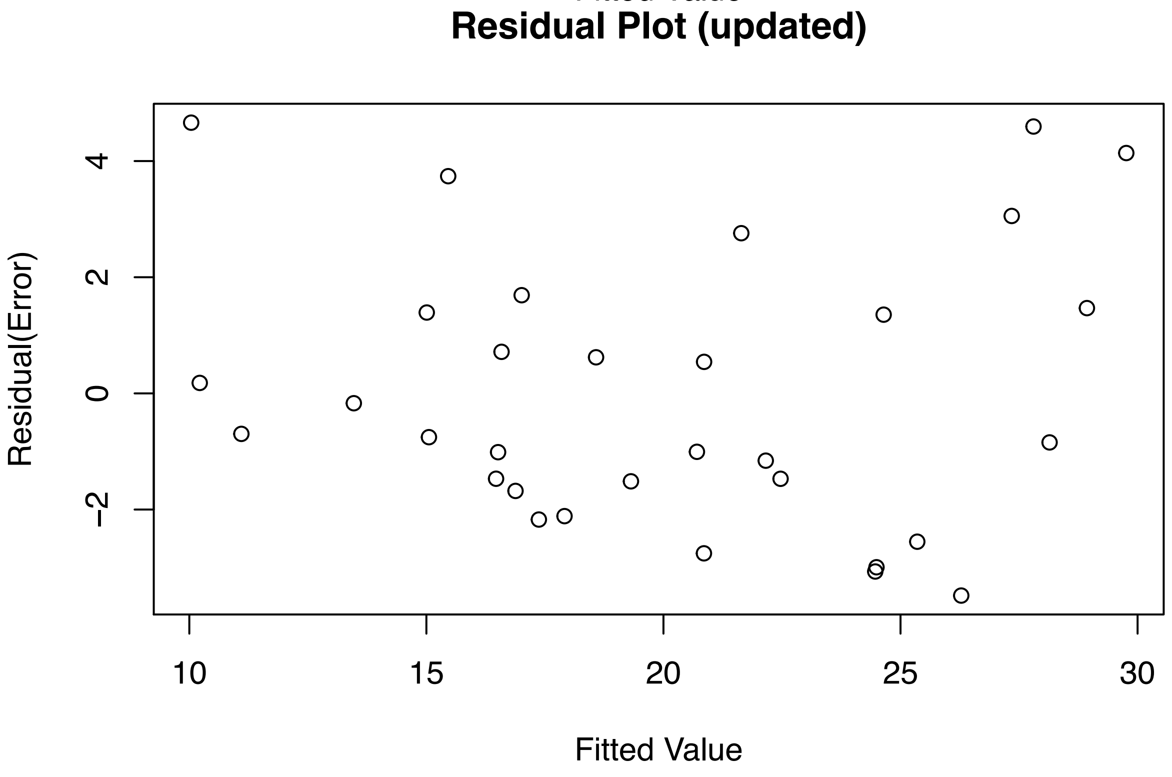
```
MSE_new <- sum((predict(result$model)-mtcars[,1])^2)/nrow(mtcars)
MSE_new
```

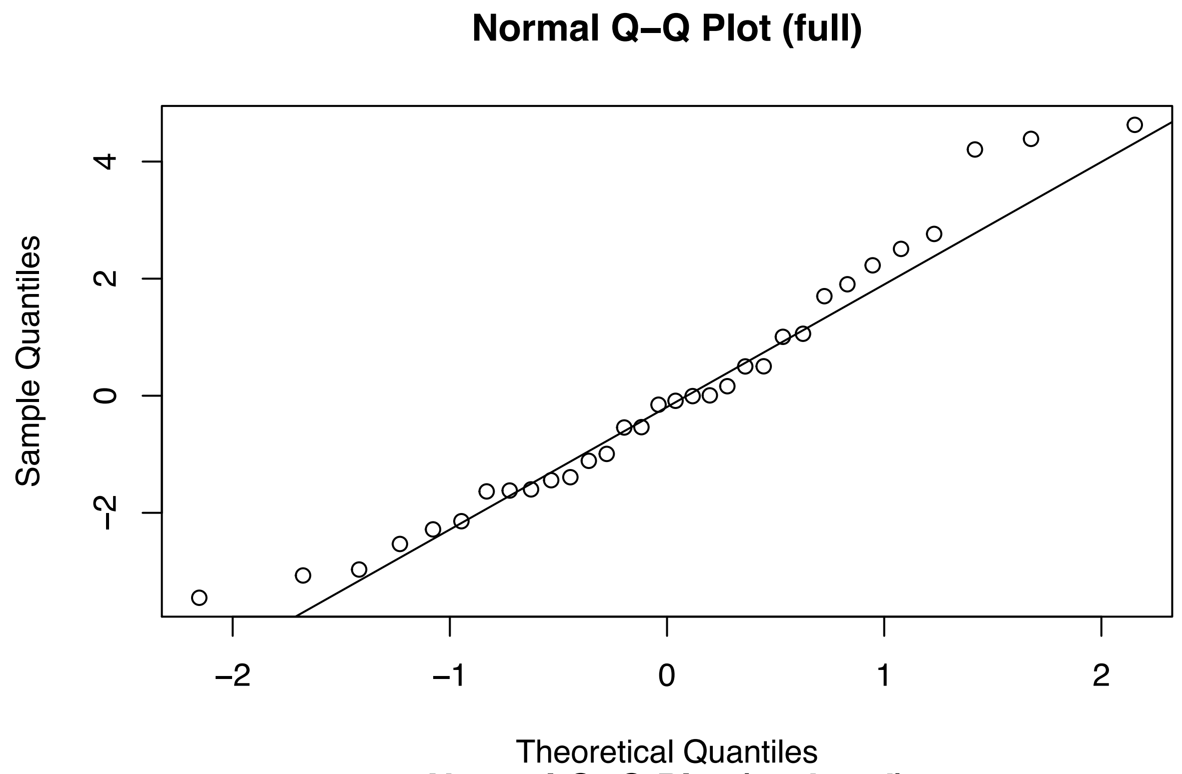
```
## [1] 5.290185
```

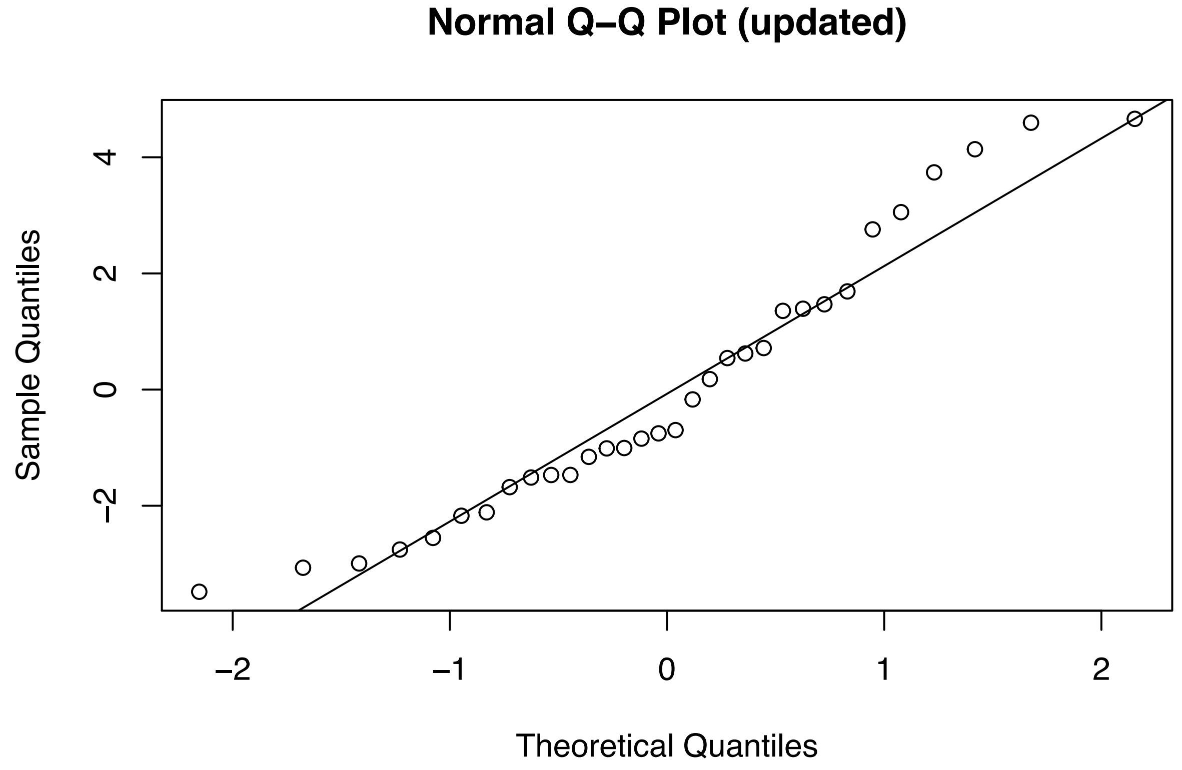
We managed to decrease the AIC at the cost of sacrificing prediction accuracy (MSE became slightly bigger).

Finally we plot to check normality of both models:









Applying genetic algorithm impairs normality of the model by introducing patterns to residuals. This is reasonable because by throwing away most of the predictors, we are trading completeness for conciseness.