Articles - Model Selection Essentials in R

Stepwise Regression Essentials in R

kassambara | 11/03/2018 | 5629 | Comment (1) | Model Selection Essentials in R

The **stepwise regression** (or stepwise selection) consists of iteratively adding and removing predictors, in the predictive model, in order to find the subset of variables in the data set resulting in the best performing model, that is a model that lowers prediction error.

There are three strategies of stepwise regression (James et al. 2014,P. Bruce and Bruce (2017)):

1. **Forward selection**, which starts with no predictors in the model, iteratively adds the most contributive predictors, and stops when the improvement is no longer statistically significant.
2. **Backward selection** (or **backward elimination**), which starts with all predictors in the model (full model), iteratively removes the least contributive predictors, and stops when you have a model where all predictors are statistically significant.
3. **Stepwise selection** (or sequential replacement), which is a combination of forward and backward selections. You start with no predictors, then sequentially add the most contributive predictors (like forward selection). After adding each new variable, remove any variables that no longer provide an improvement in the model fit (like backward selection).

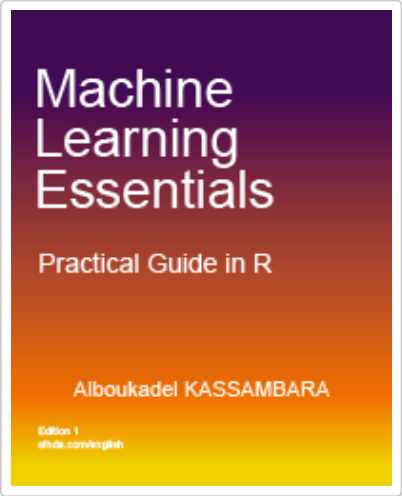
Note that,

- forward selection and stepwise selection can be applied in the high-dimensional configuration, where the number of samples n is inferior to the number of predictors p, such as in genomic fields.
- Backward selection requires that the number of samples n is larger than the number of variables p, so that the full model can be fit.

In this chapter, you'll learn how to compute the stepwise regression methods in R.

Contents:

The Book:



Machine Learning Essentials: Practical Guide in R

Loading required R packages

- `tidyverse` for easy data manipulation and visualization
- `caret` for easy machine learning workflow
- `leaps`, for computing stepwise regression

```
library(tidyverse)
library(caret)
library(leaps)
```

Computing stepwise regression

There are many functions and R packages for computing stepwise regression. These include:

- `stepAIC()` [MASS package], which choose the best model by AIC. It has an option named `direction`, which can take the following values: i) “both” (for stepwise regression, both forward and backward selection); “backward” (for backward selection) and “forward” (for forward selection). It return the best final model.

```
library(MASS)
# Fit the full model
full.model <- lm(Fertility ~., data = swiss)
```

```
# Stepwise regression model
step.model <- stepAIC(full.model, direction = "both",
                      trace = FALSE)
summary(step.model)
```

- `regsubsets()` [leaps package], which has the tuning parameter `nvmax` specifying the maximal number of predictors to incorporate in the model (See Chapter @ref(best-subsets-regression)). It returns multiple models with different size up to nvmax. You need to compare the performance of the different models for choosing the best one. `regsubsets()` has the option `method`, which can take the values "backward", "forward" and "seqrep" (seqrep = sequential replacement, combination of forward and backward selections).

```
models <- regsubsets(Fertility~., data = swiss, nvmax = 5,
                     method = "seqrep")
summary(models)
```

Note that, the `train()` function [caret package] provides an easy workflow to perform stepwise selections using the `leaps` and the MASS packages. It has an option named `method`, which can take the following values:

- `"leapBackward"`, to fit linear regression with **backward selection**
- `"leapForward"`, to fit linear regression with **forward selection**
- `"leapSeq"`, to fit linear regression with **stepwise selection** .

You also need to specify the tuning parameter `nvmax`, which corresponds to the maximum number of predictors to be incorporated in the model.

For example, you can vary `nvmax` from 1 to 5. In this case, the function starts by searching different best models of different size, up to the best 5-variables model. That is, it searches the best 1-variable model, the best 2-variables model, ..., the best 5-variables models.

The following example performs backward selection (`method = "leapBackward"`), using the `swiss` data set, to identify the best model for predicting Fertility on the basis of socio-economic indicators.

As the data set contains only 5 predictors, we'll vary `nvmax` from 1 to 5 resulting to the identification of the 5 best models with different sizes: the best 1-variable model, the best 2-variables model, ..., the best 5-variables model.

We'll use 10-fold cross-validation to estimate the average prediction error (RMSE) of each of the 5 models (see Chapter @ref(cross-validation)). The RMSE statistical metric is used to compare the 5 models and to automatically choose the best one, where best is defined as the model that minimize the RMSE.

```
# Set seed for reproducibility
set.seed(123)
# Set up repeated k-fold cross-validation
train.control <- trainControl(method = "cv", number = 10)
# Train the model
step.model <- train(Fertility ~., data = swiss,
                    method = "leapBackward",
                    tuneGrid = data.frame(nvmax = 1:5),
                    trControl = train.control
                    )
step.model$results
```

```
##   nvmax RMSE Rsquared  MAE RMSESD RsquaredSD MAESD
## 1     1 9.30    0.408 7.91   1.53      0.390  1.65
## 2     2 9.08    0.515 7.75   1.66      0.247  1.40
## 3     3 8.07    0.659 6.55   1.84      0.216  1.57
## 4     4 7.27    0.732 5.93   2.14      0.236  1.67
## 5     5 7.38    0.751 6.03   2.23      0.239  1.64
```

The output above shows different metrics and their standard deviation for comparing the accuracy of the 5 best models. Columns are:

- `nvmax`: the number of variable in the model. For example nvmax = 2, specify the best 2-variables model
- `RMSE` and `MAE` are two different metrics measuring the prediction error of each model. The lower the RMSE and MAE, the better the model.
- `Rsquared` indicates the correlation between the observed outcome values and the values predicted by the model. The higher the R squared, the better the model.

In our example, it can be seen that the model with 4 variables (nvmax = 4) is the one that has the lowest RMSE. You can display the best tuning values (nvmax), automatically selected by the `train()` function, as follow:

```
step.model$bestTune
```

```
##   nvmax
## 4     4
```

This indicates that the best model is the one with nvmax = 4 variables. The function `summary()` reports the best set of variables for each model size, up to the best 4-variables model.

```
summary(step.model$finalModel)
```

```
## Subset selection object
## 5 Variables  (and intercept)
##                  Forced in Forced out
## Agriculture          FALSE      FALSE
## Examination          FALSE      FALSE
## Education            FALSE      FALSE
## Catholic             FALSE      FALSE
## Infant.Mortality     FALSE      FALSE
## 1 subsets of each size up to 4
## Selection Algorithm: backward
##          Agriculture Examination Education Catholic Infant.Mortality
## 1  ( 1 ) " "         " "         "*"       " "      " "
## 2  ( 1 ) " "         " "         "*"       "*"      " "
## 3  ( 1 ) " "         " "         "*"       "*"      "*"
## 4  ( 1 ) "*"         " "         "*"       "*"      "*"
```

An asterisk specifies that a given variable is included in the corresponding model. For example, it can be seen that the best 4-variables model contains Agriculture, Education, Catholic, Infant.Mortality (`Fertility ~ Agriculture + Education + Catholic + Infant.Mortality`).

The regression coefficients of the final model (id = 4) can be accessed as follow:

```
coef(step.model$finalModel, 4)
```

Or, by computing the linear model using only the selected predictors:

```
lm(Fertility ~ Agriculture + Education + Catholic + Infant.Mortality,
   data = swiss)
```

```
##
## Call:
## lm(formula = Fertility ~ Agriculture + Education + Catholic +
##     Infant.Mortality, data = swiss)
##
## Coefficients:
##      (Intercept)       Agriculture         Education          Catholic
##           62.101            -0.155            -0.980             0.125
## Infant.Mortality
##            1.078
```

Discussion

This chapter describes stepwise regression methods in order to choose an optimal simple model, without compromising the model accuracy.

We have demonstrated how to use the `leaps` R package for computing stepwise regression. Another alternative is the function `stepAIC()` available in the MASS package. It has an option called `direction`, which can have the following values: "both", "forward", "backward".

```
library(MASS)
res.lm <- lm(Fertility ~., data = swiss)
step <- stepAIC(res.lm, direction = "both", trace = FALSE)
step
```

Additionally, the caret package has method to compute stepwise regression using the MASS package (`method = "lmStepAIC"`):

```
# Train the model
step.model <- train(Fertility ~., data = swiss,
                    method = "lmStepAIC",
                    trControl = train.control,
                    trace = FALSE
                    )
# Model accuracy
step.model$results
# Final model coefficients
step.model$finalModel
# Summary of the model
summary(step.model$finalModel)
```

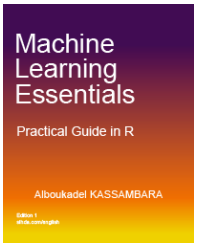
Stepwise regression is very useful for high-dimensional data containing multiple predictor variables. Other alternatives are the penalized regression (ridge and lasso regression) (Chapter @ref(penalized-regression)) and the principal components-based regression methods (PCR and PLS) (Chapter @ref(pcr-and-pls-regression)).

References

Bruce, Peter, and Andrew Bruce. 2017. *Practical Statistics for Data Scientists*. O'Reilly Media.

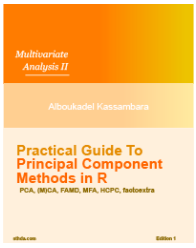
James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2014. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated.

Recommended for You!



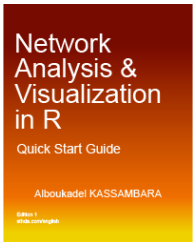
Machine Learning Essentials: Practical Guide in R

Practical Guide to Cluster Analysis in R



Practical Guide to Principal Component Methods in R

R Graphics Essentials for Great Data Visualization



Network Analysis and Visualization in R

More books on R and data science

The fields marked with a * are required !

Add a comment

Name

Visitor

Message

Preview

* Code de vérification

What is the result of 5 + ten?

Submit Reset

Visitor 05/29/2018 at 10h01
Visitor

1.Where to get p value ?
2.If i want to see any other model apart from best model ?
3. Adj R^2 ?

#504

Sign in

Login

Login

Password

Password

Auto connect

Sign in

Register

Forgotten password

Welcome!

Want to Learn More on R Programming and Data Science?
Follow us by Email

Subscribe
by FeedBurner

on Social Networks

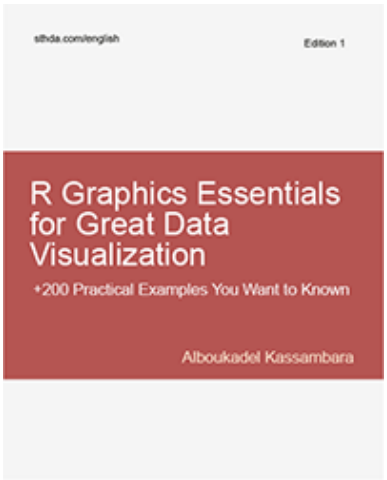
Stepwise R

Data Set Example

Data Modeling

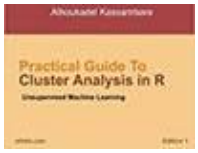
- factoextra
- survminer
- ggpubr
- ggcorrplot
- fastqcr

Our Books

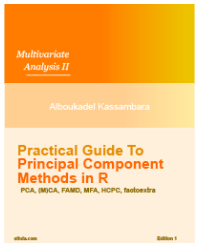


R Graphics Essentials for Great Data Visualization: 200 Practical Examples You Want to Know for Data Science

★ NEW!!

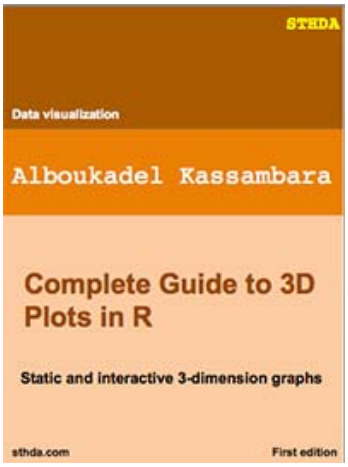


Practical Guide to Cluster Analysis in R



Practical Guide to Principal Component Methods in R

3D Plots in R



Guest Book

This site is great! I was using the PCA analysis packs FactoMineR and factoextra, and wow- what an elegant and beautiful graphic! Also, the tutorial in http://www.sthda.com/english/wiki/principal-comp... [Read more]

By *Visitor*

Guest Book

R-Bloggers

Newsletter Email