R news and tutorials contributed by (750) R bloggers

- Home
- About
- RSS
- add your blog!
- Learn R
- R jobs
- Contact us

Welcome!

R blog…

Here you will find daily **news and tutorials about R**, contributed by over 750 bloggers. There are many ways to **follow us -**

By e-mail:

Your e-mail here

Subscribe

50211 readers

BY FEEDBURNER

On Facebook:

R blog…

74K likes

Like Page

Be the first of your friends to like this

If you are an R blogger yourself you are invited to add your own R content feed to this site (**Non-English** R bloggers should add themselves- here)

Jobs for R-users

- R Developer
- postdoc in psychiatry: machine learning in human genomics
- Lead Quantitative Developer
- Research Data Analyst @ Arlington, Virginia, U.S.
- Market Research Analyst: Mobility for RSG

Popular Searches

Recent Posts

- Why R 2018 Winners
- Extracting a Reference Grid of your Data for Machine Learning Models Visualization
- #19: Intel MKL in Debian / Ubuntu follow-up
- Statistics Sunday: Converting Between Effect Sizes for Meta-Analysis
- Let R/Python send messages when the algorithms are done training
- Forecasting my weight with R
- A useful forecast combination benchmark
- A primer in using Java from R – part 1
- future.apply – Parallelize Any Base R Apply Function
- Thanks for Reading!
- A guide to working with character data in R
- Using DataCamp's Autograder to Teach R
- Melt and cast the shape of your data.frame – Exercises
- Creating Slopegraphs with R
- Parallelizing Linear Regression or Using Multiple Sources

Other sites

- SAS blogs
- Jobs for R-users

Evaluating Logistic Regression Models

August 17, 2015
By atmathew

Like 46 Share Share

(This article was first published on **Mathew Analytics » R**, and kindly contributed to R-bloggers)

Share Tweet

Logistic regression is a technique that is well suited for examining the relationship between a categorical response variable and one or more categorical or continuous predictor variables. The model is generally presented in the following format, where β refers to the parameters and

x represents the independent variables.

log(odds)=β0+β1∗x1+...+βn∗xn

The log(odds), or log-odds ratio, is defined by ln[p/(1−p)] and expresses the natural logarithm of the ratio between the probability that an event will occur, p(Y=1), to the probability that it will not occur. We are usually concerned with the predicted probability of an event occuring and that is defined by p=1/1+exp^−z, where z=β0+β1∗x1+...+βn∗xn

Logistic Regression Example

We will use the *GermanCredit* dataset in the *caret* package for this example. It contains *62* characteristics and *1000*observations, with a target variable (*Class*) that is allready defined. The response variable is coded *0* for bad consumer and *1* for good. It's always recommended that one looks at the coding of the response variable to ensure that it's a factor variable that's coded accurately with a *0/1* scheme or two factor levels in the right order. The first step is to partition the data into training and testing sets.

```
library(caret)
data(GermanCredit)

Train <- createDataPartition(GermanCredit$Class, p=0.6, list=FALSE)
training <- GermanCredit[ Train, ]
testing <- GermanCredit[ -Train, ]
```

Using the training dataset, which contains *600* observations, we will use logistic regression to model *Class* as a function of five predictors.

```
mod_fit <- train(Class ~ Age + ForeignWorker + Property.RealEstate + Housing.Own +
                   CreditHistory.Critical,  data=training, method="glm", family="binomial")
```

Bear in mind that the estimates from logistic regression characterize the relationship between the predictor and response variable on a log-odds scale. For example, this model suggests that for every one unit increase in *Age*, the log-odds of the consumer having good credit increases by 0.018. Because this isn't of much practical value, we'll ussually want to use the exponential function to calculate the odds ratios for each preditor.

```
exp(coef(mod_fit$finalModel))
```

```
##             (Intercept)                     Age          ForeignWorker
##               1.1606762               1.0140593               0.5714748
##    Property.RealEstate             Housing.Own CreditHistory.Critical
##               1.8214566               1.6586940               2.5943711
```

This informs us that for every one unit increase in *Age*, the odds of having good credit increases by a factor of 1.01. In many cases, we often want to use the model parameters to predict the value of the target variable in a completely new set of observations. That can be done with the predict function. Keep in mind that if the model was created using the glm function, you'll need to add `type="response"` to the predict command.

```
predict(mod_fit, newdata=testing)
predict(mod_fit, newdata=testing, type="prob")
```

Model Evaluation and Diagnostics

A logistic regression model has been built and the coefficients have been examined. However, some critical questions remain. Is the model any good? How well does the model fit the data? Which predictors are most important? Are the predictions accurate? The rest of this document will cover techniques for answering these questions and provide R code to conduct that analysis.

For the following sections, we will primarily work with the logistic regression that I created with the `glm()` function. While I prefer utilizing the `Caret` package, many functions in `R` will work better with a `glm` object.

```
mod_fit_one <- glm(Class ~ Age + ForeignWorker + Property.RealEstate + Housing.Own +
                     CreditHistory.Critical, data=training, family="binomial")

mod_fit_two <- glm(Class ~ Age + ForeignWorker, data=training, family="binomial")
```

Goodness of Fit

Likelihood Ratio Test

A logistic regression is said to provide a better fit to the data if it demonstrates an improvement over a model with fewer predictors. This is performed using the likelihood ratio test, which compares the likelihood of the data under the full model against the likelihood of the data under a model with fewer predictors. Removing predictor variables from a model will almost always make the model fit less well (i.e. a model will have a lower log likelihood), but it is necessary to test whether the observed difference in model fit is statistically significant. Given that H0 holds that the reduced model is true, a p-value for the overall model fit statistic that is less than 0.05 would compel us to reject the null hypothesis. It would provide evidence against the reduced model in favor of the current model. The likelihood ratio test can be performed in R using the `lrtest()` function from the `lmtest` package or using the `anova()` function in base.

```
anova(mod_fit_one, mod_fit_two, test ="Chisq")
```

```
library(lmtest)
lrtest(mod_fit_one, mod_fit_two)
```

Pseudo R^2

Unlike linear regression with ordinary least squares estimation, there is no R2 statistic which explains the proportion of variance in the dependent variable that is explained by the predictors. However, there are a number of pseudo R2 metrics that could be of value. Most notable is McFadden's R2, which is defined as 1−[ln(LM)/ln(L0)] where ln(LM) is the log likelihood value for the fitted model and ln(L0) is the log likelihood for the null model with only an intercept as a predictor. The measure ranges from *0* to just under *1*, with values closer to zero indicating that the model has no predictive power.

```
library(pscl)

pR2(mod_fit_one)  # look for 'McFadden'
```

```
##           llh       llhNull            G2      McFadden          r2ML
## -344.42107079 -366.51858123   44.19502089    0.06029029    0.07101099
##          r2CU
##    0.10068486
```

Hosmer-Lemeshow Test

Another approch to determining the goodness of fit is through the Homer-Lemeshow statistics, which is computed on data after the observations have been segmented into groups based on having similar predicted probabilities. It examines whether the observed proportions of events are similar to the predicted probabilities of occurence in subgroups of the data set using a pearson chi square test. Small values with large p-values indicate a good fit to the data while large values with p-values below 0.05 indicate a poor fit. The null hypothesis holds that the model fits the data and in the below example we would reject H0.

```
library(MKmisc)
HLgof.test(fit = fitted(mod_fit_one), obs = training$Class)

library(ResourceSelection)
hoslem.test(training$Class, fitted(mod_fit_one), g=10)
```

Statistical Tests for Individual Predictors

Wald Test

A wald test is used to evaluate the statistical significance of each coefficient in the model and is calculated by taking the ratio of the square of the regression coefficient to the square of the standard error of the coefficient. The idea is to test the hypothesis that the coefficient of an independent variable in the model is significantly different from zero. If the test fails to reject the null hypothesis, this suggests that removing the variable from the model will not substantially harm the fit of that model.

```
library(survey)

regTermTest(mod_fit_one, "ForeignWorker")
```

```
## Wald test for ForeignWorker
##  in glm(formula = Class ~ Age + ForeignWorker + Property.RealEstate +
##     Housing.Own + CreditHistory.Critical, family = "binomial",
```

```
##     data = training)
## F =  0.949388  on  1  and  594  df: p= 0.33027

regTermTest(mod_fit_one, "CreditHistory.Critical")

## Wald test for CreditHistory.Critical
##  in glm(formula = Class ~ Age + ForeignWorker + Property.RealEstate +
##     Housing.Own + CreditHistory.Critical, family = "binomial",
##     data = training)
## F =  16.67828  on  1  and  594  df: p= 5.0357e-05
```

Variable Importance

To assess the relative importance of individual predictors in the model, we can also look at the absolute value of the t-statistic for each model parameter. This technique is utilized by the varImp function in the caret package for general and generalized linear models.

```
varImp(mod_fit)

## glm variable importance
##
##                          Overall
## CreditHistory.Critical  100.00
## Property.RealEstate      57.53
## Housing.Own              50.73
## Age                      22.04
## ForeignWorker             0.00
```

Validation of Predicted Values

Classification Rate

When developing models for prediction, the most critical metric regards how well the model does in predicting the target variable on out of sample observations. The process involves using the model estimates to predict values on the training set. Afterwards, we will compared the predicted target variable versus the observed values for each observation. In the example below, you'll notice that our model accurately predicted 67 of the observations in the testing set.

```
pred = predict(mod_fit, newdata=testing)
accuracy <- table(pred, testing[,"Class"])
sum(diag(accuracy))/sum(accuracy)

## [1] 0.705

pred = predict(mod_fit, newdata=testing)
confusionMatrix(data=pred, testing$Class)
```

ROC Curve

The receiving operating characteristic is a measure of classifier performance. Using the proportion of positive data points that are correctly considered as positive and the proportion of negative data points that are mistakenly considered as positive, we generate a graphic that shows the trade off between the rate at which you can correctly predict something with the rate of incorrectly predicting something. Ultimately, we're concerned about the area under the ROC curve, or AUROC. That metric ranges from 0.50 to 1.00, and values above 0.80 indicate that the model does a good job in discriminating between the two categories which comprise our target variable. Bear in mind that ROC curves can examine both target-x-predictor pairings and target-x-model performance. An example of both are presented below.

```
library(pROC)
# Compute AUC for predicting Class with the variable CreditHistory.Critical
f1 = roc(Class ~ CreditHistory.Critical, data=training)
plot(f1, col="red")

##
## Call:
## roc.formula(formula = Class ~ CreditHistory.Critical, data = training)
##
## Data: CreditHistory.Critical in 180 controls (Class Bad) < 420 cases (Class Good).
## Area under the curve: 0.5944

library(ROCR)
# Compute AUC for predicting Class with the model
prob <- predict(mod_fit_one, newdata=testing, type="response")
pred <- prediction(prob, testing$Class)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(perf)

auc <- performance(pred, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.6540625
```

K-Fold Cross Validation

When evaluating models, we often want to assess how well it performs in predicting the target variable on different subsets of the data. One such technique for doing this is k-fold cross-validation, which partitions the data into k equally sized segments (called 'folds'). One fold is held out for validation while the other k-1 folds are used to train the model and then used to predict the target variable in our testing data. This process is repeated k times, with the performance of each model in predicting the hold-out set being tracked using a performance metric such as accuracy. The most common variation of cross validation is 10-fold cross-validation.

```
ctrl <- trainControl(method = "repeatedcv", number = 10, savePredictions = TRUE)

mod_fit <- train(Class ~ Age + ForeignWorker + Property.RealEstate + Housing.Own +
                   CreditHistory.Critical,  data=GermanCredit, method="glm", family="binomial",
                 trControl = ctrl, tuneLength = 5)

pred = predict(mod_fit, newdata=testing)
confusionMatrix(data=pred, testing$Class)
```

There you have it. A high level review of evaluating logistic regression models in R. If you have any feedback or suggestions, please comment in the section below.

Share Tweet

To **leave a comment** for the author, please follow the link and comment on their blog: **Mathew Analytics » R**.

R-bloggers.com offers **daily e-mail updates** about R news and tutorials on topics such as: Data science, Big Data, R jobs, visualization (ggplot2, Boxplots, maps, animation), programming (RStudio, Sweave, LaTeX, SQL, Eclipse, git, hadoop, Web Scraping) statistics (regression, PCA, time series, trading) and more...

If you got this far, why not **subscribe for updates** from the site? Choose your flavor: e-mail, twitter, RSS, or facebook...

Like 46 Share Share

Comments are closed.

Search R-bloggers

Google Custom Search

Recent popular posts

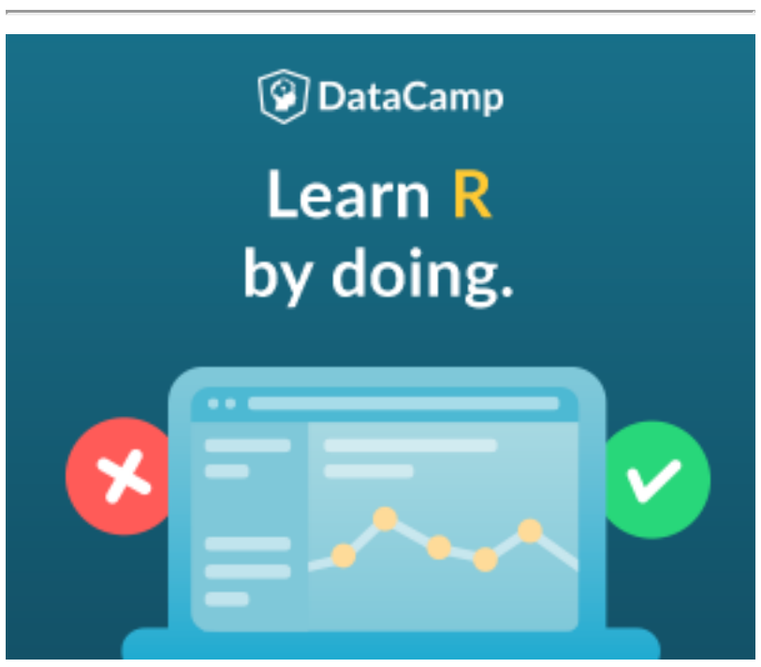
- future.apply - Parallelize Any Base R Apply Function
- Let R/Python send messages when the algorithms are done training
- A primer in using Java from R - part 1
- Forecasting my weight with R
- Why R 2018 Winners

Most visited articles of the week

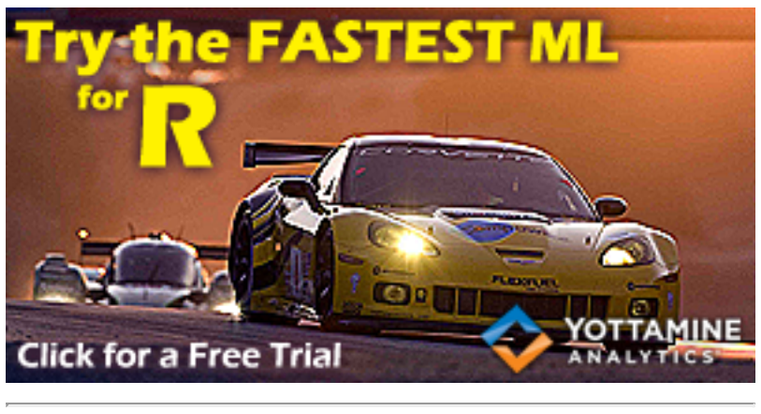
1. How to write the first for loop in R
2. Installing R packages
3. Using apply, sapply, lapply in R
4. R – Sorting a data frame by the contents of a column
5. How to perform a Logistic Regression in R
6. How to Make a Histogram with Basic R

7. Tutorials for learning R
8. How to Make a Histogram with ggplot2
9. Creating Slopegraphs with R

Sponsors



Quantide: statistical consulting and training

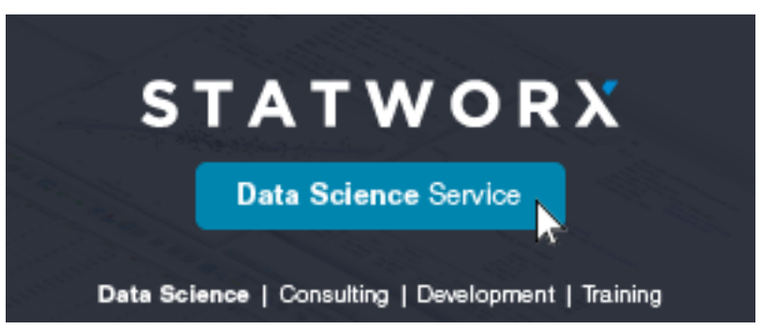


Our ads respect your privacy. Read our Privacy Policy page to learn more.

Contact us if you wish to help support R-bloggers, and place **your banner here**.

QUANTIDE
knowledge from data

Quantide: statistical consulting and training



Our ads respect your privacy. Read our
Privacy Policy page to learn more.

Contact us if you wish to help support R-bloggers, and place **your banner here**.

Jobs for R users

- R Developer
- postdoc in psychiatry: machine learning in human genomics
- Lead Quantitative Developer
- Research Data Analyst @ Arlington, Virginia, U.S.
- Market Research Analyst: Mobility for RSG
- Data Scientist @ New Delhi, India
- Data Scientist/Programmer @ Milwaukee, Wisconsin, U.S.

Full list of contributing R-bloggers
R-bloggers was founded by Tal Galili, with gratitude to the R community.
Is powered by WordPress using a bavotasan.com design.
Copyright © 2018 **R-bloggers**. All Rights Reserved. Terms and Conditions for this website