r-statistics.co (/) by Selva Prabhakaran

Search..

-

Tutorial

R Tutorial (R-Tutorial.html)

ggplot2

ggplot2 Short Tutorial (ggplot2-Tutorial-With-R.html)

ggplot2 Tutorial 1 - Intro (Complete-Ggplot2-Tutorial-Part1-With-R-Code.html)

ggplot2 Tutorial 2 - Theme (Complete-Ggplot2-Tutorial-Part2-Customizing-Theme-With-R-Code.html)

ggplot2 Tutorial 3 - Masterlist (Top50-Ggplot2-Visualizations-MasterList-R-Code.html)

ggplot2 Quickref (ggplot2-cheatsheet.html)

Foundations

Linear Regression (Linear-Regression.html)

Statistical Tests (Statistical-Tests-in-R.html)

Missing Value Treatment (Missing-Value-Treatment-With-R.html)

Outlier Analysis (Outlier-Treatment-With-R.html)

Feature Selection (Variable-Selection-and-Importance-With-R.html)

Model Selection (Model-Selection-in-R.html)

Logistic Regression (Logistic-Regression-With-R.html)

Advanced Linear Regression (Environments.html)

Advanced Regression Models

Advanced Regression Models (adv-regression-models.html)

Time Series

Time Series Analysis (Time-Series-Analysis-With-R.html)

Time Series Forecasting (Time-Series-Forecasting-With-R.html)

More Time Series Forecasting (Time-Series-Forecasting-With-R-part2.html)

High Performance Computing

Parallel computing (Parallel-Computing-With-R.html)

Strategies to Speedup R code (Strategies-To-Improve-And-Speedup-R-Code.html)

Useful Techniques

Association Mining (Association-Mining-With-R.html)

Multi Dimensional Scaling (Multi-Dimensional-Scaling-With-R.html)

Optimization (Profiling.html)

InformationValue package (Information-Value-With-R.html)

Contents

> Ordinal logistic regression can be used to model a ordered factor response.

The `polr()` function from the `MASS` package can be used to build the *proportional odds logistic regression* and predict the class of multi-class ordered variables. One such use case is described below.

Example: Predict Cars Evaluation

Below is a example on how we can use ordered logistic regression to predict the cars evaluation based on cars evaluation dataset (http://archive.ics.uci.edu/ml/datasets/Car+Evaluation). The cars are evaluated as one amongst `very good`, `good`, `acceptable` or `unacceptable`. The attributes of the cars available to use to predict this decision are:

1. buying : v-high, high, med, low
2. maint : v-high, high, med, low
3. doors : 2, 3, 4, 5-more
4. persons : 2, 4, more
5. lug_boot : small, med, big
6. safety : low, med, high

Also, it is worthwhile to note that about 70% of the cars are evaluated as *unacceptable*. The class distribution of the ordered multi class *Y* is as follows:

class	N	N[%]
unacc	1210	(70.023 %)
acc	384	(22.222 %)
good	69	(3.993 %)
v-good	65	(3.762 %)

Lets being the modeling process by first importing the data and assigning the correct orders to the factor variables.

Import the data

```
carsdata <- read.csv("http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data", header=F, stringsAsFactors=F)  # import string variables as characters.
colnames(carsdata) <- c("buying", "maint", "doors", "persons", "lug_boot", "safety", "class")
```

Reorder the levels of factors

In order logistic regression, the order of the levels in the factor variables matters. So, lets define them explicitly. This is an critical step, otherwise, predictions could go worng easily.

```
# Reorder
carsdata$buying <- factor(carsdata$buying, levels=c("low", "med", "high", "vhigh"), orde
red=TRUE)
carsdata$maint <- factor(carsdata$maint, levels=c("low", "med", "high", "vhigh"), ordere
d=TRUE)
carsdata$doors <- factor(carsdata$doors, levels=c("2", "3", "4", "5more"), ordered=TRUE)
carsdata$persons <- factor(carsdata$persons, levels=c("2", "4", "more"), ordered=TRUE)
carsdata$lug_boot <- factor(carsdata$lug_boot, levels=c("small", "med", "big"), ordered=
TRUE)
carsdata$safety <- factor(carsdata$safety, levels=c("low", "med", "high"), ordered=TRUE)
carsdata$class <- factor(carsdata$class, levels=c("unacc", "acc", "good", "vgood"), orde
red=TRUE)
```

Prepare training and test data

```
# Prepare Training and Test Data
set.seed(100)
trainingRows <- sample(1:nrow(carsdata), 0.7 * nrow(carsdata))
trainingData <- carsdata[trainingRows, ]
testData <- carsdata[-trainingRows, ]
```

Build the ordered logistic regression model

```
### Build ordered logistic regression model
options(contrasts = c("contr.treatment", "contr.poly"))
polrMod <- polr(class ~ safety + lug_boot + doors + buying + maint, data=trainingData)
summary(polrMod)
#> Call:
#> polr(formula = class ~ safety + lug_boot + doors + buying + maint,
#>     data = trainingData)
#>
#> Coefficients:
#>               Value Std. Error   t value
#> safety.L    19.9443    0.06145  324.5411
#> safety.Q   -10.6548    0.10088 -105.6189
#> lug_boot.L   1.0119    0.14011    7.2224
#> lug_boot.Q  -0.3197    0.13355   -2.3940
#> doors.L      0.5415    0.15573    3.4774
#> doors.Q     -0.2787    0.15466   -1.8018
#> doors.C     -0.1096    0.15372   -0.7132
#> buying.L    -2.0945    0.18137  -11.5480
#> buying.Q    -0.1369    0.15659   -0.8746
#> buying.C     0.5219    0.15318    3.4069
#> maint.L     -1.8209    0.17533  -10.3856
#> maint.Q     -0.4768    0.15811   -3.0153
#> maint.C      0.3319    0.15518    2.1388
#>
#> Intercepts:
#>             Value     Std. Error t value
#> unacc|acc     9.4557    0.0740   127.8297
#> acc|good     11.8726    0.1345    88.2882
#> good|vgood   13.1331    0.1997    65.7533
#>
#> Residual Deviance: 1300.15
#> AIC: 1332.15
```

Predict on test data

```
### Predict
predictedClass <- predict(polrMod, testData)  # predict the classes directly
head(predictedClass)
#> [1] unacc unacc unacc unacc unacc unacc
#> Levels: unacc acc good vgood

predictedScores <- predict(polrMod, testData, type="p")  # predict the probabilites
head(predictedScores)
#>        unacc          acc         good        vgood
#> 3  0.9774549 2.049194e-02 1.470224e-03 5.829671e-04
#> 6  0.9347665 5.904708e-02 4.424660e-03 1.761744e-03
#> 12 0.9774549 2.049194e-02 1.470224e-03 5.829671e-04
#> 13 1.0000000 3.574918e-14 2.664535e-15 8.881784e-16
#> 14 0.9762376 2.159594e-02 1.551314e-03 6.151902e-04
#> 18 0.9120030 7.946377e-02 6.099087e-03 2.434191e-03

## Confusion matrix and misclassification error

table(testData$class, predictedClass)  # confusion matrix
#>        predictedClass
#>       unacc acc good vgood
#> unacc   305  45    0     4
#> acc      60  60    0     0
#> good      0  17    0     0
#> vgood     0  18    0    10

mean(as.character(testData$class) != as.character(predictedClass))  # misclassification
 error
#> 0.277
```