Data 621 - Blog 4

Leticia Salazar

November 27th, 2022

Contents

Generalized Linear Model:

We have so far created simple and multiple linear models; now I will be introducing generalized linear model. This is a generalization of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution.

GLM build a linear relationship between the response and predictors even if their relationship is not linear. There are some basic assumptions for GLMs (some are modified for LMs):

- Data should be independent and random (each random variable has the same probability distribution)
- The response variable `y` does not need to be normally distributed, but the distribution is from an exponential family (e.g. bionmia, Poisson, multinomial, normal)
- The original response variable need not have a linear relationship with the independent variables, but the transformed response variable (through the link function) is linearly dependent on the independent variables.
- Feature engineering on the Independent variable can be applied i.e instead of taking the original raw independent variables, variable transformation can be done, and the transformed independent variables, such as taking a log transformation, squaring the variables, reciprocal of the variables, can also be used to build the GLM model.
- Homoscedasticity (i.e constant variance) need not be satisfied. Response variable Error variance can increase, or decrease with the independent variables.
- Errors are independent but need not be normally distributed

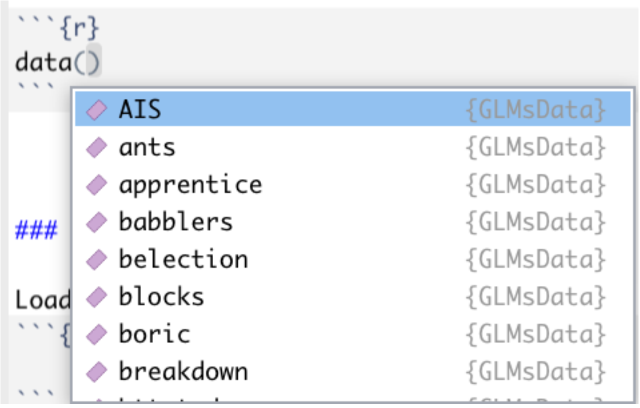
To start we will be using a data from the 'GLMsData' cran r-project package to select a data set to create our generalized linear models.

Load Libraries

These are the libraries we will be using:

```
library(ggplot2)
library(tidyverse)
library(jtools)
library(hrbrthemes)
library(GLMsData)
library(gtsummary)
library(MASS)
library(performance)
```

There's a couple of data sets that we can use from the `GLMsData` but in order to view these options on your end use the function `data()` and you'll see a small box with the list of the data sets in this library.



~~~
```{r}
data()
```
~~~

Load Data

We will be using the Australian Institute of Sports (AIS) data, which has physical and blood measurements from high performance athletes at the AIS.

Name:	Description:
Sex	the sex of the athlete: F means female, and M means male
Sport	the sport of the athlete; one of BBall (basketball), Field, Gym (gymnastics), Netball, Rowing, Swim (swimming), T400m, (track, further than 400m), Tennis, TPSprnt (track sprint events), WPolo (waterpolo)
LBM	lean body mass, in kg
Ht	height, in cm
Wt	weight, in kg
BMI	body mass index, in kg per metre-squared
SSF	sum of skin folds
PBF	percentage body fat
RBC	red blood cell count, in 1012 per litre

Name:	Description:
WBC	white blood cell count, in 1012 per litre
HCT	hematocrit, in percent
HGB	hemoglobin concentration, in grams per decilitre
Ferr	plasma ferritins, in ng per decilitre

To load the data we the `data()` function and the `head()` function to view it:

```
##   Sex Sport   LBM    Ht   Wt   BMI   SSF  RBC WBC  HCT  HGB Ferr   PBF
## 1   F BBall 63.32 195.9 78.9 20.56 109.1 3.96 7.5 37.5 12.3   60 19.75
## 2   F BBall 58.55 189.7 74.4 20.67 102.8 4.41 8.3 38.2 12.7   68 21.30
## 3   F BBall 55.36 177.8 69.1 21.86 104.6 4.14 5.0 36.4 11.6   21 19.88
## 4   F BBall 57.18 185.0 74.9 21.88 126.4 4.11 5.3 37.3 12.6   69 23.66
## 5   F BBall 53.20 184.6 64.6 18.96  80.3 4.45 6.8 41.5 14.0   29 17.64
## 6   F BBall 53.77 174.0 63.7 21.04  75.2 4.10 4.4 37.4 12.5   42 15.58
```

Data Exploration

Lets get the structure of this data; Using the `str()` function we notice we have 202 observations with 13 variables.

```
## 'data.frame':    202 obs. of  13 variables:
##  $ Sex  : Factor w/ 2 levels "F","M": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Sport: Factor w/ 10 levels "BBall","Field",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ LBM  : num  63.3 58.5 55.4 57.2 53.2 ...
##  $ Ht   : num  196 190 178 185 185 ...
##  $ Wt   : num  78.9 74.4 69.1 74.9 64.6 63.7 75.2 62.3 66.5 62.9 ...
##  $ BMI  : num  20.6 20.7 21.9 21.9 19 ...
##  $ SSF  : num  109.1 102.8 104.6 126.4 80.3 ...
##  $ RBC  : num  3.96 4.41 4.14 4.11 4.45 4.1 4.31 4.42 4.3 4.51 ...
##  $ WBC  : num  7.5 8.3 5 5.3 6.8 4.4 5.3 5.7 8.9 4.4 ...
##  $ HCT  : num  37.5 38.2 36.4 37.3 41.5 37.4 39.6 39.9 41.1 41.6 ...
##  $ HGB  : num  12.3 12.7 11.6 12.6 14 12.5 12.8 13.2 13.5 12.7 ...
##  $ Ferr : int  60 68 21 69 29 42 73 44 41 44 ...
##  $ PBF  : num  19.8 21.3 19.9 23.7 17.6 ...
```

Using `summary()` function to get the statistical structure of our data

```
##  Sex         Sport         LBM               Ht              Wt
##  F:100   Rowing :37   Min.   : 34.36   Min.   :148.9   Min.   : 37.80
##  M:102   T400m  :29   1st Qu.: 54.67   1st Qu.:174.0   1st Qu.: 66.53
##          BBall  :25   Median : 63.03   Median :179.7   Median : 74.40
##          Netball:23   Mean   : 64.87   Mean   :180.1   Mean   : 75.01
##          Swim   :22   3rd Qu.: 74.75   3rd Qu.:186.2   3rd Qu.: 84.12
##          Field  :19   Max.   :106.00   Max.   :209.4   Max.   :123.20
##          (Other):47
##       BMI             SSF              RBC             WBC
##  Min.   :16.75   Min.   : 28.00   Min.   :3.800   Min.   : 3.300
##  1st Qu.:21.08   1st Qu.: 43.85   1st Qu.:4.372   1st Qu.: 5.900
##  Median :22.72   Median : 58.60   Median :4.755   Median : 6.850
##  Mean   :22.96   Mean   : 69.02   Mean   :4.719   Mean   : 7.109
##  3rd Qu.:24.46   3rd Qu.: 90.35   3rd Qu.:5.030   3rd Qu.: 8.275
##  Max.   :34.42   Max.   :200.80   Max.   :6.720   Max.   :14.300
##
##       HCT             HGB             Ferr             PBF
##  Min.   :35.90   Min.   :11.60   Min.   :  8.00   Min.   : 5.630
##  1st Qu.:40.60   1st Qu.:13.50   1st Qu.: 41.25   1st Qu.: 8.545
##  Median :43.50   Median :14.70   Median : 65.50   Median :11.650
##  Mean   :43.09   Mean   :14.57   Mean   : 76.88   Mean   :13.507
##  3rd Qu.:45.58   3rd Qu.:15.57   3rd Qu.: 97.00   3rd Qu.:18.080
##  Max.   :59.70   Max.   :19.20   Max.   :234.00   Max.   :35.520
##
```

And of course use of `gtsummary()` library to view our data's structure differently.

```
## Table printed with ‘knitr::kable()‘, not {gt}. Learn why at
## https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html
## To suppress this message, include ‘message = FALSE‘ in code chunk header.
```

Characteristic	**N = 202**
Sex	
F	100 (50%)
M	102 (50%)
Sport	
BBall	25 (12%)
Field	19 (9.4%)
Gym	4 (2.0%)
Netball	23 (11%)
Rowing	37 (18%)
Swim	22 (11%)
T400m	29 (14%)
Tennis	11 (5.4%)
TSprnt	15 (7.4%)
WPolo	17 (8.4%)
LBM	63 (55, 75)
Ht	180 (174, 186)
Wt	74 (67, 84)
BMI	22.72 (21.08, 24.46)
SSF	59 (44, 90)
RBC	4.76 (4.37, 5.03)
WBC	6.85 (5.90, 8.28)
HCT	43.5 (40.6, 45.6)
HGB	14.70 (13.50, 15.57)
Ferr	66 (41, 97)
PBF	11.7 (8.5, 18.1)

Model Building:

To create our models we use the following syntax:

glm (formula, family, data, weights, subset, Start=null, model=TRUE,method=” ”. . .)

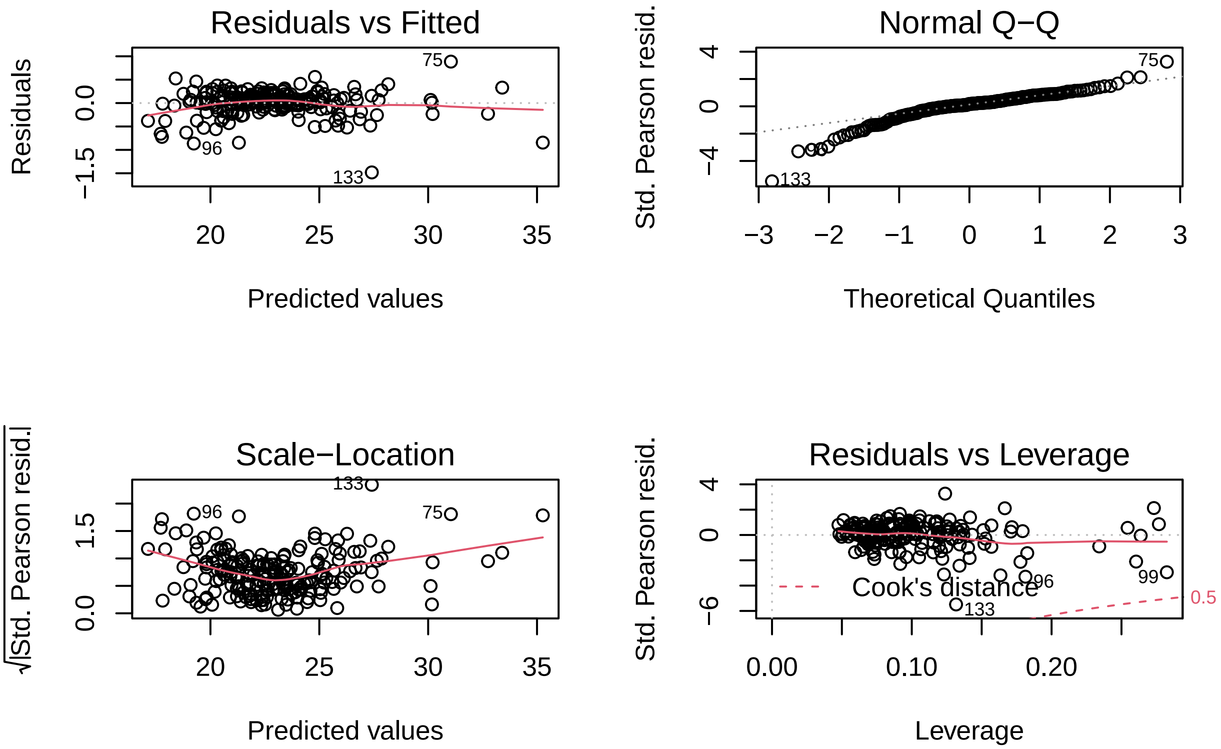
Below we have other link functions to use depending on the family you will be using:

Family	Default Link Function
binomial	(link = “logit”)
gaussian	(link = “identity”)
Gamma	(link = “inverse”)
inverse.gaussian	(link = “1/muˆ2”)
poisson	(link = “log”)
quasi	(link = “identity”, variance = “constant”)
quasibinomial	(link = “logit”)
quasipoisson	(link = “log”)

I have opted to used the guassian family for the models below.

Model 1 will be a base model:

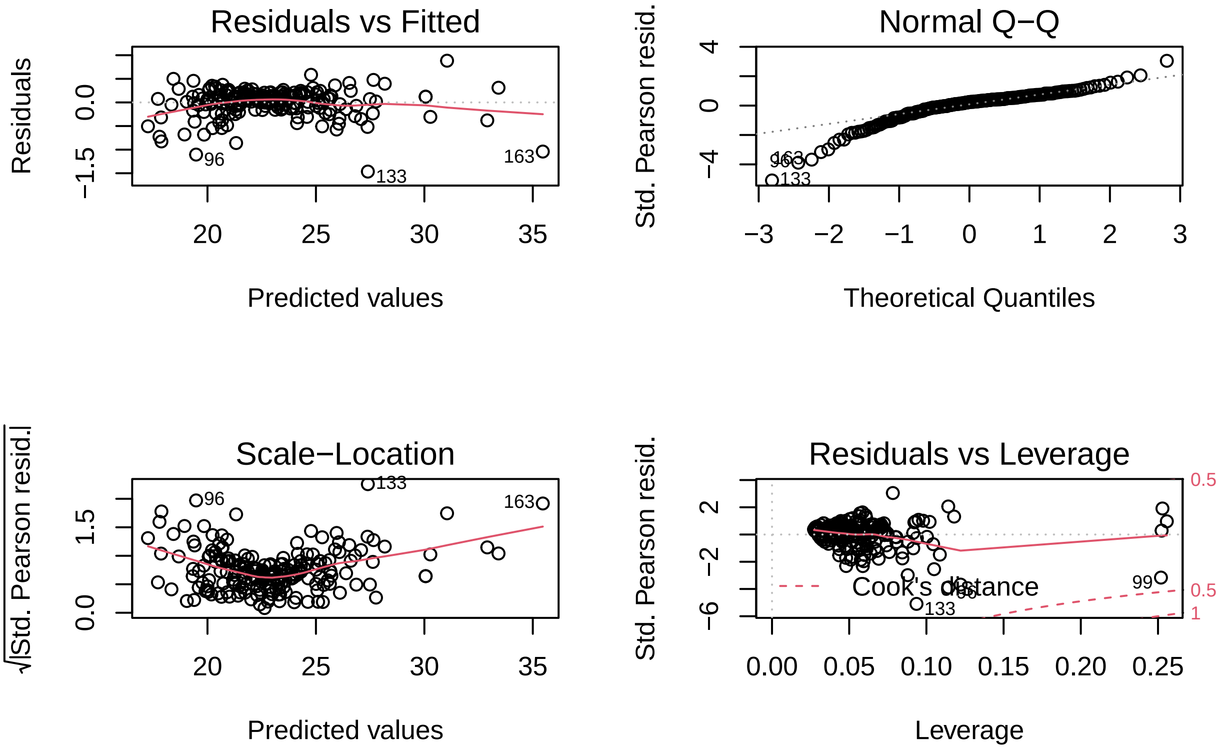
```
##
## Call:
## glm(formula = BMI ~ Sex + Sport + LBM + Ht + Wt + SSF + PBF +
##     WBC + HCT + HGB + Ferr, family = gaussian(link = "identity"),
##     data = AIS)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.48202  -0.09002   0.04780   0.16179   0.88570
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  42.5942420  0.9111177  46.749  < 2e-16 ***
## SexM          0.1922183  0.1134696   1.694  0.09197 .
## SportField    0.3233329  0.1152391   2.806  0.00557 **
## SportGym     -1.2811581  0.1964370  -6.522 6.64e-10 ***
## SportNetball  0.0600713  0.0977253   0.615  0.53952
## SportRowing   0.1005367  0.0818731   1.228  0.22105
## SportSwim     0.2194978  0.0976737   2.247  0.02582 *
## SportT400m    0.1421550  0.1051004   1.353  0.17787
## SportTennis  -0.0186392  0.1195542  -0.156  0.87628
## SportTSprnt   0.2977256  0.1206913   2.467  0.01456 *
## SportWPolo    0.0179661  0.1046757   0.172  0.86391
## LBM           0.0344505  0.0371041   0.928  0.35439
## Ht           -0.2378588  0.0047731 -49.833  < 2e-16 ***
## Wt            0.2624938  0.0327839   8.007 1.35e-13 ***
## SSF           0.0033669  0.0034438   0.978  0.32953
## PBF           0.0411612  0.0314146   1.310  0.19176
## WBC           0.0066557  0.0129736   0.513  0.60856
## HCT          -0.0122763  0.0193140  -0.636  0.52582
## HGB           0.0542869  0.0542895   1.000  0.31866
## Ferr         -0.0001934  0.0005264  -0.367  0.71370
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.08399842)
##
##     Null deviance: 1648.624  on 201  degrees of freedom
## Residual deviance:   15.288  on 182  degrees of freedom
## AIC: 93.845
##
## Number of Fisher Scoring iterations: 2
```



For Model2 we use Backwards elimination:

```
##
## Call:
## glm(formula = BMI ~ Sport + Ht + Wt, family = gaussian(link = "identity"),
##     data = AIS)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.46437  -0.10602   0.07047   0.15836   0.88324
##
## Coefficients:
```

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  44.694272   0.650160  68.743  < 2e-16 ***
## SportField    0.211174   0.111202   1.899   0.0591 .
## SportGym     -1.505624   0.190622  -7.898 2.21e-13 ***
## SportNetball  0.156607   0.094649   1.655   0.0997 .
## SportRowing   0.063430   0.081893   0.775   0.4396
## SportSwim     0.077695   0.091998   0.845   0.3994
## SportT400m   -0.020106   0.090648  -0.222   0.8247
## SportTennis  -0.104362   0.116670  -0.895   0.3722
## SportTSprnt   0.110755   0.105509   1.050   0.2952
## SportWPolo   -0.001183   0.097734  -0.012   0.9904
## Ht           -0.246724   0.004324 -57.063  < 2e-16 ***
## Wt            0.302235   0.003130  96.556  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.09119712)
##
##     Null deviance: 1648.624  on 201  degrees of freedom
## Residual deviance:   17.327  on 190  degrees of freedom
## AIC: 103.14
##
## Number of Fisher Scoring iterations: 2
```



Based on the summary output and plots of the two models, model2 has a higher AIC hence seeming to be a better fit model.

When working with different data you'd be able to perform more than 2 models along with checking performance and making predictions on the evaluation data. Yet this is the start to perform generalized linear models.

References:

- Team, D. C. (2020, June 30). GLM in R: Generalized linear model tutorial. DataCamp. Retrieved November 27, 2022, from https://www.datacamp.com/tutorial/generalized-linear-models
- Comprehensive R Archive Network (CRAN). (n.d.). Package glmsdata. CRAN. Retrieved November 27, 2022, from https://cran.r-project.org/web/packages/GLMsData/
- Team, G. L. (2022, October 27). Generalized linear model: What does it mean? Great Learning Blog: Free Resources what Matters to shape your Career! Retrieved November 27, 2022, from https://www.mygreatlearning.com/blog/generalized-linear-models/
- robk@statmethods.net, R. K.-. (n.d.). Generalized linear models. Quick-R: Generalized Linear Models. Retrieved November 27, 2022, from https://www.statmethods.net/advstats/glm.html