

A tutorial on fitting discrete response models in high-dimensional datasets with the `countgmifs` package

Kellie J. Archer^{*}, Rebecca R. Lehman^a, Mateusz Makowski^b

^{*} The Ohio State University, ^a The United Network of Organ Sharing, ^b The EMMES Corporation

Abstract

In this tutorial we describe our `countgmifs` R package, available from the Comprehensive R Archive Network, that can fit Poisson and negative binomial models when the number of predictors (P) exceeds the sample size (N). We then illustrate the functions in the `countgmifs` R package using simulated data.

Keywords: discrete response, high-dimensional features, penalized models, R.

1. Introduction

Various algorithms can be used for obtaining solutions for the Least Absolute Shrinkage and Selection Operator (LASSO) (Tibshirani 1996, 1997) and elastic net penalized models (Zou and Hastie 2005). In the linear regression setting, the Incremental Forward Stagewise (IFS) is a penalized solution that enforces monotonicity (Hastie *et al.* 2007). IFS can be generalized to problems involving other than squared error loss, and the adaption is called the generalized monotone incremental forward stagewise (GMIFS) method (Hastie *et al.* 2007). Herein we extended the GMIFS method (Hastie *et al.* 2007) to the discrete response setting and implemented functions in our `countgmifs` R package (Makowski and Archer 2015). The `countgmifs` function can be used to fit penalized Poisson and negative binomial regression models in the presence of a high-dimensional covariate space.

2. Implementation

The `countgmifs` package was written in the R programming environment (R Core Team 2018). The `countgmifs` function allows the user to specify a model formula, identify the matrix of covariates to be penalized in the model fitting algorithm using the `x` parameter, and additionally specify the model type (`family`) as either "poisson" or "nb" (default). The defaults for updating the penalized coefficients are `epsilon=0.001` and `tol=1e-5`. Our likelihood functions were written in R and tested by comparing our R output to output produced by the `glm` function and `glm.nb` function in the R MASS package using benchmark datasets for data where $P < N$.

3. Example

We will generate data where the outcome follows a negative binomial distribution to illustrate our `countgmifs` package. First, we set the random seed simply so that our results can be replicated.

```
> set.seed(26)
```

Next, we set the sample size to $N = 50$, the number of covariates to $P = 500$, the intercept to $\beta_0 = 0.50$, the true parameter values are set to $\pm \log(1.5)$ for the first 5 variables and to 0 for the remaining 495 predictors, and the heterogeneity parameter to $\alpha = 0.50$.

```
> n <- 50
> p <- 500
> intercept <- .5
> beta <- c(log(1.5), log(1.5), -log(1.5), -log(1.5), -log(1.5), rep(0,495))
> alpha <- 0.5
```

Using these settings, we generate our P covariates from a standard normal distribution, calculate the observation mean μ_i , then generate the discrete response from the negative binomial distribution using this mean and our heterogeneity parameter α . Thereafter we combine the discrete response y with the matrix of covariates x to form our data frame.

```
> x <- matrix(rnorm(n*p,0,1), nrow=n, ncol=p, byrow=TRUE)
> colnames(x) <- paste("Var", 1:p, sep="")
> mu <- exp(intercept + crossprod(t(x), beta))
> y <- rnbino(n=n, size=1/alpha, mu=mu) # Discrete response
> data <- data.frame(y, x)
```

Now we can fit our negative binomial GMIFS model. We load the library then make a call to the `countgmifs` function.

```
> library("countgmifs")
> nb <- countgmifs(y ~ 1, data=data, offset=NULL, x=x, epsilon=0.01, tol=0.001,
+               scale=TRUE, verbose=FALSE)
```

To fit a model where all predictors are penalized the model formula is specified to fit an intercept only model and the predictors to be penalized are specified using the `x` parameter. Because `offset=NULL`, that indicates we are modeling a count rather than a rate, where the denominator for the rate would be passed using the `offset` parameter. The parameter `x` specifies the variables that are included in the model in a “penalized” fashion. The `x` parameter can either be a vector naming columns in the `data.frame` specified by the `data` parameter or `x` can be the `data.frame` name with the columns to include (or exclude) indicated by their (negative) index. Prior to model fitting NA values should be imputed or removed from the `data.frame`. By default, `epsilon=0.001`, however, to reduce processing time for CRAN testing builds of this package, we have changed `epsilon=0.01` and `tol=0.001`. The parameter `scale=TRUE` indicates that the predictors are to be centered and scaled; `verbose=FALSE` indicates we do not want to monitor the step in the fitting process. Because the GMIFS procedure is incremental, the user may want to specify `verbose=TRUE` to print the step number

in order to monitor the status of the model fitting procedure. By default a negative binomial regression model is fit; a Poisson model can be fit by specifying `family="poisson"`.

Methods including `coef`, `plot`, `predict`, `fitted`, `print`, and `summary` can be applied to `countgmifs` model objects. Because the returned list differs depending on whether a no penalty subset is included or a Poisson versus a negative binomial model is fit, the `print` function returns the object names of the fitted object.

```
> print(nb)
```

```
[1] "a"      "beta"   "theta"  "x"      "y"      "scale"  "logLik"
[8] "AIC"    "BIC"    "w"      "offset" "family"
```

By default `coef`, `predict`, and `summary` extracts the relevant information from the step in the solution path that attained the minimum BIC.

```
> summary(nb)
```

```
nb model
```

```
alpha      = 1.064559
at step    = 66
logLik     = -93.21502
AIC        = 198.43
BIC        = 209.9022
```

	alpha (Intercept)	Var1	Var2	Var3
	1.0645594	0.6567073	0.0000000	0.0000000
	Var4	Var5	Var6	Var7
	-0.3400000	0.0000000	0.0000000	0.0000000
	Var9	Var10	Var11	Var12
	0.0000000	0.0000000	0.0000000	0.0000000
	Var14	Var15	Var16	Var17
	0.0000000	0.0000000	0.0000000	0.0000000
	Var19	Var20	Var21	Var22
	0.0000000	0.0000000	0.0000000	0.0000000
	Var24	Var25	Var26	Var27
	0.0000000	0.0000000	0.0000000	0.0000000
	Var29	Var30	Var31	Var32
	0.0000000	0.0000000	0.0000000	0.0000000
	Var34	Var35	Var36	Var37
	0.0000000	0.0000000	0.0000000	0.0000000
	Var39	Var40	Var41	Var42
	0.0000000	0.0000000	0.0000000	0.0000000
	Var44	Var45	Var46	Var47
	0.0000000	0.0000000	0.0000000	0.0000000
	Var49	Var50	Var51	Var52
	0.0000000	0.0000000	0.0000000	0.0000000
	Var54	Var55	Var56	Var57
				Var58

0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var59	Var60	Var61	Var62	Var63
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var64	Var65	Var66	Var67	Var68
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var69	Var70	Var71	Var72	Var73
0.0000000	0.0000000	0.0000000	0.0000000	0.2200000
Var74	Var75	Var76	Var77	Var78
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var79	Var80	Var81	Var82	Var83
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var84	Var85	Var86	Var87	Var88
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var89	Var90	Var91	Var92	Var93
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var94	Var95	Var96	Var97	Var98
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var99	Var100	Var101	Var102	Var103
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var104	Var105	Var106	Var107	Var108
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var109	Var110	Var111	Var112	Var113
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var114	Var115	Var116	Var117	Var118
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var119	Var120	Var121	Var122	Var123
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var124	Var125	Var126	Var127	Var128
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var129	Var130	Var131	Var132	Var133
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var134	Var135	Var136	Var137	Var138
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var139	Var140	Var141	Var142	Var143
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var144	Var145	Var146	Var147	Var148
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var149	Var150	Var151	Var152	Var153
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var154	Var155	Var156	Var157	Var158
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var159	Var160	Var161	Var162	Var163
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var164	Var165	Var166	Var167	Var168
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var169	Var170	Var171	Var172	Var173
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000

Var174	Var175	Var176	Var177	Var178
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var179	Var180	Var181	Var182	Var183
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var184	Var185	Var186	Var187	Var188
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var189	Var190	Var191	Var192	Var193
0.0000000	0.0000000	0.0000000	0.0000000	0.0700000
Var194	Var195	Var196	Var197	Var198
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var199	Var200	Var201	Var202	Var203
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var204	Var205	Var206	Var207	Var208
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var209	Var210	Var211	Var212	Var213
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var214	Var215	Var216	Var217	Var218
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var219	Var220	Var221	Var222	Var223
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var224	Var225	Var226	Var227	Var228
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var229	Var230	Var231	Var232	Var233
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var234	Var235	Var236	Var237	Var238
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var239	Var240	Var241	Var242	Var243
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var244	Var245	Var246	Var247	Var248
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var249	Var250	Var251	Var252	Var253
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var254	Var255	Var256	Var257	Var258
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var259	Var260	Var261	Var262	Var263
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var264	Var265	Var266	Var267	Var268
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var269	Var270	Var271	Var272	Var273
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var274	Var275	Var276	Var277	Var278
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var279	Var280	Var281	Var282	Var283
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var284	Var285	Var286	Var287	Var288
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var289	Var290	Var291	Var292	Var293

0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var294	Var295	Var296	Var297	Var298
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var299	Var300	Var301	Var302	Var303
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var304	Var305	Var306	Var307	Var308
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var309	Var310	Var311	Var312	Var313
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var314	Var315	Var316	Var317	Var318
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var319	Var320	Var321	Var322	Var323
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var324	Var325	Var326	Var327	Var328
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var329	Var330	Var331	Var332	Var333
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var334	Var335	Var336	Var337	Var338
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var339	Var340	Var341	Var342	Var343
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var344	Var345	Var346	Var347	Var348
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var349	Var350	Var351	Var352	Var353
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var354	Var355	Var356	Var357	Var358
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var359	Var360	Var361	Var362	Var363
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var364	Var365	Var366	Var367	Var368
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var369	Var370	Var371	Var372	Var373
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var374	Var375	Var376	Var377	Var378
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var379	Var380	Var381	Var382	Var383
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var384	Var385	Var386	Var387	Var388
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var389	Var390	Var391	Var392	Var393
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var394	Var395	Var396	Var397	Var398
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var399	Var400	Var401	Var402	Var403
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var404	Var405	Var406	Var407	Var408
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000

Var409	Var410	Var411	Var412	Var413
-0.0200000	0.0000000	0.0000000	0.0000000	0.0000000
Var414	Var415	Var416	Var417	Var418
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var419	Var420	Var421	Var422	Var423
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var424	Var425	Var426	Var427	Var428
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var429	Var430	Var431	Var432	Var433
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var434	Var435	Var436	Var437	Var438
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var439	Var440	Var441	Var442	Var443
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var444	Var445	Var446	Var447	Var448
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var449	Var450	Var451	Var452	Var453
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var454	Var455	Var456	Var457	Var458
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var459	Var460	Var461	Var462	Var463
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var464	Var465	Var466	Var467	Var468
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var469	Var470	Var471	Var472	Var473
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var474	Var475	Var476	Var477	Var478
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var479	Var480	Var481	Var482	Var483
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var484	Var485	Var486	Var487	Var488
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var489	Var490	Var491	Var492	Var493
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var494	Var495	Var496	Var497	Var498
0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
Var499	Var500			
0.0000000	0.0000000			

However, any step along the solution path can be extracted by specifying the step using the `model.select` parameter for these three functions. For example, the model attaining the minimum BIC can be extracted using `summary(nb, model.select="AIC")`.

Alternatively, the 250th step can be extracted using `summary(nb, model.select=250)`.

The `plot` function plots the solution path of the model fit. The vertical axis can be changed using the `type` parameter with allowable selections being `"trace"` (default), `"AIC"`, `"BIC"` or

"logLik". Although there are default x-axis, y-axis, and titles provided for each plot, the user can modify these by supplying their own arguments to `xlab`, `ylab`, and `main`, respectively.

```
> plot(nb)
```

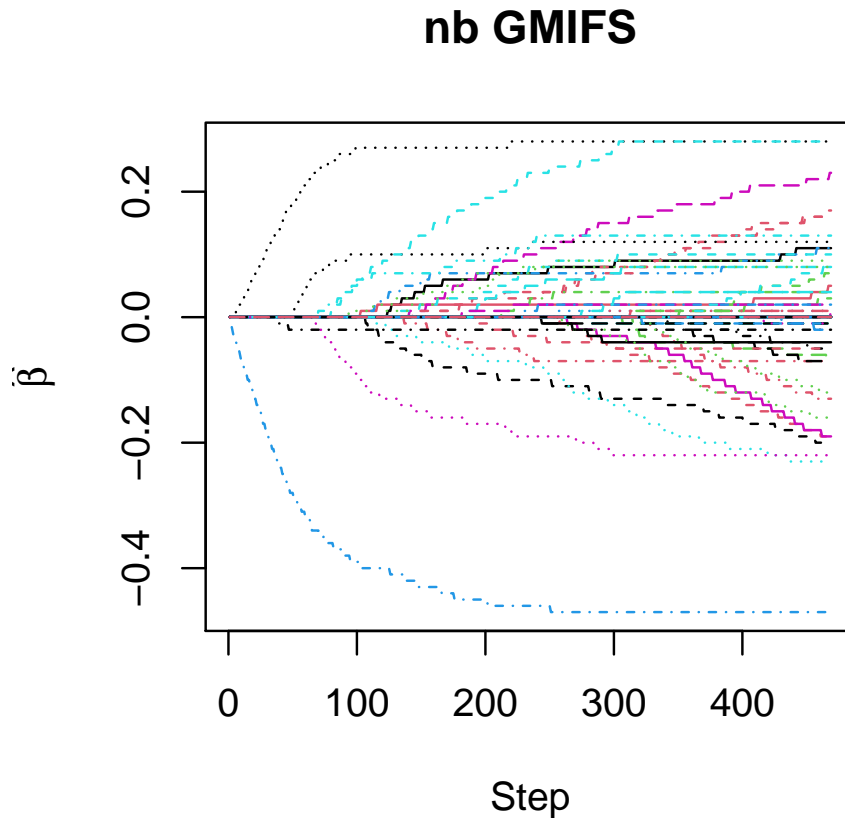


Figure 1: Coefficient estimates along the solution path for a fitted `countgmifs` object.

The `coef` function extracts the estimated parameters and returns them as a vector. The default is to select the coefficients from the step at which the BIC attains a minimum.

```
> coef.BIC <- coef(nb)
> coef.BIC[coef.BIC != 0]
```

alpha (Intercept)	Var4	Var73	Var193
1.0645594	0.6567073	-0.3400000	0.2200000
Var409			0.0700000
-0.0200000			

We can also extract the coefficients from the AIC selected model.


```
> coef.AIC<-coef(nb, model.select="AIC")
> coef.AIC[coef.AIC!=0]
```

```

      alpha  (Intercept)          Var1          Var4          Var26
0.0008423968 0.4157010990 0.0800000000 -0.4700000000 0.0200000000
      Var30          Var32          Var34          Var73          Var92
0.1100000000 0.0400000000 0.0700000000 0.2800000000 0.0400000000
      Var99          Var157          Var164          Var182          Var193
0.0400000000 -0.1100000000 -0.0700000000 -0.0400000000 0.1200000000
      Var215          Var271          Var299          Var300          Var304
0.0600000000 -0.0100000000 0.1300000000 0.0200000000 0.0200000000
      Var325          Var347          Var383          Var393          Var409
-0.0100000000 0.0800000000 -0.0900000000 0.0800000000 -0.0200000000
      Var437          Var483          Var498
0.2400000000 0.0900000000 -0.1900000000
```

The `predict` function (or equivalently, `fitted`) returns the fitted values from the model. As with `coef` and `summary` the `predict` function by default extracts the model that attained the minimum BIC, but predictions for any step along the solution path can be obtained by specifying the step using the `model.select` parameter.

```
> yhat <- predict(nb, model.select="AIC")
```

The National Institutes of Health released notice NOT-OD-15-102 detailing the requirement for researchers to consider sex as a biological variable, which may lead the analyst to coerce sex into the multivariable model. There are a multitude of clinical scenarios where it is of primary interest to discover the additional predictive value of including molecular features beyond already known risk factors. Therefore, we extended our method to penalize some covariates (high-throughput genomic features) without penalizing others (such as demographic and/or clinical covariates) [Gentry et al. \(2015\)](#). The following example is merely to illustrate additional flexibility of the package. Suppose that `Var1` is to be coerced into the model while `Var2 - Var10` are to be penalized. Any variable(s) to be coerced into the model should appear on the right-hand side of the model formula and represents the “unpenalized predictors.” The model can be fit using

```
> nb.2<-countgmifs(y ~ Var1 , data=data, offset=NULL,
+   x=c("Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8", "Var9", "Var10"),
+   epsilon=0.01, tol=0.001, scale=TRUE, verbose=FALSE)
> summary(nb.2)
```

```
nb model
alpha      = 0.2548902
at step    = 187
logLik     = -81.74046
AIC        = 179.4809
BIC        = 194.7771
```

```
> plot(yhat, y)
```

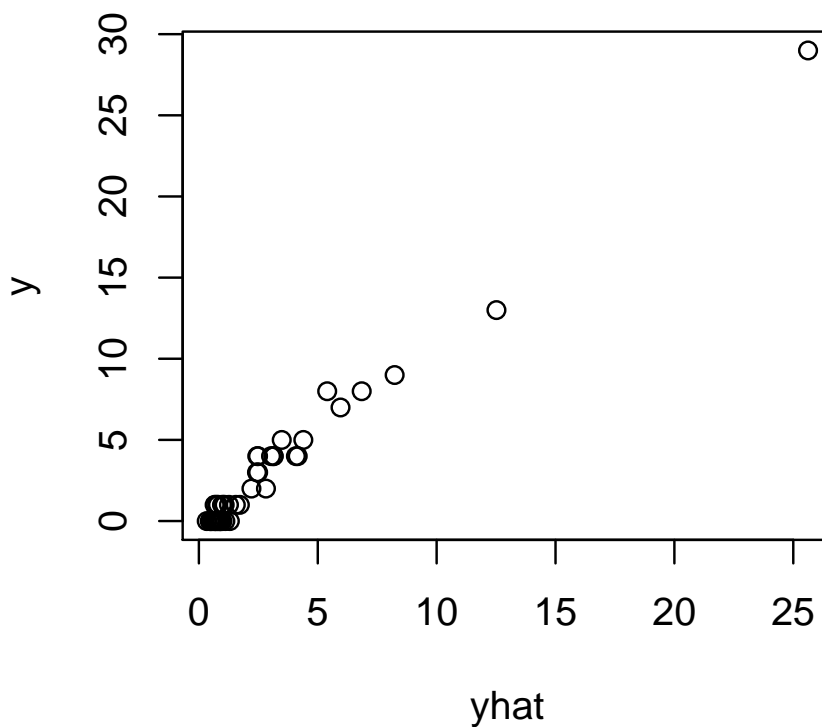


Figure 2: Observed versus fitted values from the `countgmifs` object.

alpha (Intercept)		Var1	Var2	Var3
0.2548902	0.3587775	0.5455997	0.4400000	-0.3200000
Var4	Var5	Var6	Var7	Var8
-0.8600000	-0.1400000	0.0000000	0.0000000	0.0000000
Var9	Var10			
0.0000000	-0.1000000			

When predicting the outcome for a new set of observations, the `predict` function will accept arguments for `newx` (the penalized predictors), `neww` (the unpenalized predictors using model formula notation), `newdata` (new `data.frame` that contains the unpenalized predictors), and `newoffset` which is required if an offset was used in the fitted model. Suppose we want to leave observation 1 out, fit the model, then predict the class for observation 1 as a left out test set where we have coerced `Var1` in the model by including it as an unpenalized predictor. The following code would be used:

```
nb.m1 <- countgmifs(y ~ Var1 , data=data[-1,], offset=NULL,
  x=c("Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8", "Var9", "Var10"),
  epsilon=0.01, tol=0.001, scale=TRUE, verbose=FALSE)
predict(nb.m1, neww=~Var1, newdata=data[1,],
newx=data[1,c("Var2", "Var3", "Var4", "Var5", "Var6", "Var7", "Var8", "Var9", "Var10")])
```

Acknowledgments

Research reported in this tutorial was supported by the National Library Of Medicine of the National Institutes of Health under Award Number R01LM011169 and by the National Institute of Environmental Health Sciences under Award Number T32ES007334. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

- Gentry A, Jackson-Cook C, Lyon D, Archer K (2015). “Penalized Ordinal Regression Methods for Predicting Stage of Cancer in High-Dimensional Covariate Spaces.” *Cancer Informatics*, **14(Suppl 2)**, 201–208.
- Hastie T, Taylor J, Tibshirani R, Walther G (2007). “Forward stagewise regression and the monotone lasso.” *Electronic Journal of Statistics*, **1**, 1–29.
- Makowski M, Archer K (2015). “Generalized monotone incremental forward stagewise method for modeling count data: Application predicting micronuclei frequency.” *Cancer Informatics*, **14**, 97?105.
- R Core Team (2018). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.
- Tibshirani R (1996). “Regression shrinkage and selection via the Lasso.” *Journal of the Royal Statistical Society, B*, **58**, 267–288.
- Tibshirani R (1997). “The lasso method for variable selection in the Cox model.” *Statistics in Medicine*, **16**, 385–395.
- Zou H, Hastie T (2005). “Regularization and variable selection via the elastic net.” *Journal of the Royal Statistical Society B*, **67**, 301–320.

Affiliation:

Kellie J. Archer
 Division of Biostatistics
 College of Public Health
 The Ohio State University

1841 Neil Ave.
240 Cunz Hall
Columbus, OH 43210
E-mail: archer.43@osu.edu
URL: <https://cph.osu.edu/people/karcher>

Rebecca R. Lehman
United Network for Organ Sharing
Richmond, VA
E-mail: lehmanrr@mymail.vcu.edu

Mateusz Makowski
The EMMES Corporation
401 N. Washington Street
Rockville, MD 20850
E-mail: mmakowski@emmes.com