Bayesian Logistic Regression

Eric C. Brown

September 23, 2013

1 Model

$$\mathbf{y} \sim \text{Bern}(\boldsymbol{\theta})$$
$$\boldsymbol{\theta} = \frac{1}{1 + \exp(-\boldsymbol{\mu})}$$
$$\boldsymbol{\mu} = \beta_0 + \mathbf{X}\boldsymbol{\beta}$$
$$\beta_0 \sim \mathcal{N}(0, 100)$$
$$\beta_j \sim \mathcal{N}(0, 100), \quad j = 1, \ldots, J$$

2 Code

```
bayesianLogisticRegression <- function(y, X = NULL, Xnew = NULL, iter = 5000,
    thin = 1, chains = 1, burnin = floor(iter/2), betameanpriors = if (is.null(X)) {
        0
    } else {
        rep(0, ncol(X) + 1)
    }, betasdpriors = if (is.null(X)) {
        100
    } else {
        rep(100, ncol(X) + 1)
    }, inits = NULL, seed = 1) {
    if (is.null(X)) {
        J <- 0
    } else {
        J <- ncol(X)
        if (!(nrow(X) == length(y))) {
            message("ERROR: X and y must have the same number of rows.")
            message(paste(c("ERROR: nrow(X) =", nrow(X), ", length(y) =", length(y)),
                collapse = " "))
            return(NA)
        }
    }
    if (!(is.null(Xnew))) {
        if (!(ncol(Xnew) == J)) {
            message("ERROR: X and Xnew must have the same number of columns.")
            message(paste(c("ERROR: ncol(X) =", ncol(X), ", ncol(Xnew) =", ncol(Xnew)),
                collapse = " "))
            return()
        }
```

```
    }
    if (is.null(X)) {
        form <- formula("y ~ 1")
        mm <- model.matrix(form, data.frame(y = y))
    } else {
        form <- formula(paste("y ~ ", paste(names(X), collapse = "+"), sep = ""))
        mm <- model.matrix(form, X)
    }
    if (is.null(inits)) {
        fit <- glm.fit(mm, y, family = binomial(link = "logit"))
        inits <- coef(fit)
    }
    if (is.null(Xnew)) {
        model <- "\n  data {\n    int<lower=0>              N;\n    int<lower=1>              J;\n    in
        data <- list(N = nrow(mm), J = ncol(mm), y = y, X = mm, betameanpriors = betameanpriors,
            betasdpriors = betasdpriors)
        set.seed(seed)
        print(seed)
        fit <- stan(model_code = model, data = data, chains = chains, iter = iter,
            warmup = burnin, thin = thin, init = function(x) {
                list(beta = inits)
            }, nondiag_mass = TRUE, seed = seed)
    } else {
        mmnew <- model.matrix(form, Xnew)
        model <- "\n  data {\n    int<lower=1>               N;\n    int<lower=1>               J;\n
        data <- list(N = nrow(mm), J = ncol(mm), y = y, X = mm, betameanpriors = betameanpriors,
            betasdpriors = betasdpriors, Nnew = nrow(mmnew), Xnew = mmnew)
        set.seed(seed)
        print(seed)
        fit <- stan(model_code = model, data = data, chains = chains, iter = iter,
            warmup = burnin, thin = thin, init = function(x) {
                list(beta = inits)
            }, nondiag_mass = TRUE, seed = seed)
    }
    return(fit)
}
```

3 Example

Predict whether the car will have an automatic (0) or manual (1) transmission.

```
xtable(mtcars, caption = "\\textbf{mtcars} data set.")
```

```
y <- mtcars[, "am"]
X <- as.data.frame(mtcars[, c("mpg", "cyl", "disp", "hp", "drat", "wt", "vs",
    "gear", "carb")])
Xnew <- X
fit <- bayesianLogisticRegression(y = y, X = X, iter = 10000, chains = 4, inits = rep(0,
    ncol(X) + 1))

## [1] 1
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.00	6.00	160.00	110.00	3.90	2.62	16.46	0.00	1.00	4.00	4.00
Mazda RX4 Wag	21.00	6.00	160.00	110.00	3.90	2.88	17.02	0.00	1.00	4.00	4.00
Datsun 710	22.80	4.00	108.00	93.00	3.85	2.32	18.61	1.00	1.00	4.00	1.00
Hornet 4 Drive	21.40	6.00	258.00	110.00	3.08	3.21	19.44	1.00	0.00	3.00	1.00
Hornet Sportabout	18.70	8.00	360.00	175.00	3.15	3.44	17.02	0.00	0.00	3.00	2.00
Valiant	18.10	6.00	225.00	105.00	2.76	3.46	20.22	1.00	0.00	3.00	1.00
Duster 360	14.30	8.00	360.00	245.00	3.21	3.57	15.84	0.00	0.00	3.00	4.00
Merc 240D	24.40	4.00	146.70	62.00	3.69	3.19	20.00	1.00	0.00	4.00	2.00
Merc 230	22.80	4.00	140.80	95.00	3.92	3.15	22.90	1.00	0.00	4.00	2.00
Merc 280	19.20	6.00	167.60	123.00	3.92	3.44	18.30	1.00	0.00	4.00	4.00
Merc 280C	17.80	6.00	167.60	123.00	3.92	3.44	18.90	1.00	0.00	4.00	4.00
Merc 450SE	16.40	8.00	275.80	180.00	3.07	4.07	17.40	0.00	0.00	3.00	3.00
Merc 450SL	17.30	8.00	275.80	180.00	3.07	3.73	17.60	0.00	0.00	3.00	3.00
Merc 450SLC	15.20	8.00	275.80	180.00	3.07	3.78	18.00	0.00	0.00	3.00	3.00
Cadillac Fleetwood	10.40	8.00	472.00	205.00	2.93	5.25	17.98	0.00	0.00	3.00	4.00
Lincoln Continental	10.40	8.00	460.00	215.00	3.00	5.42	17.82	0.00	0.00	3.00	4.00
Chrysler Imperial	14.70	8.00	440.00	230.00	3.23	5.34	17.42	0.00	0.00	3.00	4.00
Fiat 128	32.40	4.00	78.70	66.00	4.08	2.20	19.47	1.00	1.00	4.00	1.00
Honda Civic	30.40	4.00	75.70	52.00	4.93	1.61	18.52	1.00	1.00	4.00	2.00
Toyota Corolla	33.90	4.00	71.10	65.00	4.22	1.83	19.90	1.00	1.00	4.00	1.00
Toyota Corona	21.50	4.00	120.10	97.00	3.70	2.46	20.01	1.00	0.00	3.00	1.00
Dodge Challenger	15.50	8.00	318.00	150.00	2.76	3.52	16.87	0.00	0.00	3.00	2.00
AMC Javelin	15.20	8.00	304.00	150.00	3.15	3.44	17.30	0.00	0.00	3.00	2.00
Camaro Z28	13.30	8.00	350.00	245.00	3.73	3.84	15.41	0.00	0.00	3.00	4.00
Pontiac Firebird	19.20	8.00	400.00	175.00	3.08	3.85	17.05	0.00	0.00	3.00	2.00
Fiat X1-9	27.30	4.00	79.00	66.00	4.08	1.94	18.90	1.00	1.00	4.00	1.00
Porsche 914-2	26.00	4.00	120.30	91.00	4.43	2.14	16.70	0.00	1.00	5.00	2.00
Lotus Europa	30.40	4.00	95.10	113.00	3.77	1.51	16.90	1.00	1.00	5.00	2.00
Ford Pantera L	15.80	8.00	351.00	264.00	4.22	3.17	14.50	0.00	1.00	5.00	4.00
Ferrari Dino	19.70	6.00	145.00	175.00	3.62	2.77	15.50	0.00	1.00	5.00	6.00
Maserati Bora	15.00	8.00	301.00	335.00	3.54	3.57	14.60	0.00	1.00	5.00	8.00
Volvo 142E	21.40	4.00	121.00	109.00	4.11	2.78	18.60	1.00	1.00	4.00	2.00

Table 1: **mtcars** data set.

```
##
## TRANSLATING MODEL 'model' FROM Stan CODE TO C++ CODE NOW.
## COMPILING THE C++ CODE FOR MODEL 'model' NOW.
## /Users/brown/.R/Makevars:123: warning: overriding commands for target `.c.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:123: warning: ignoring old commands
## /Users/brown/.R/Makevars:125: warning: overriding commands for target `.c.d'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:125: warning: ignoring old commands
## /Users/brown/.R/Makevars:128: warning: overriding commands for target `.cc.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:128: warning: ignoring old commands
## /Users/brown/.R/Makevars:130: warning: overriding commands for target `.cpp.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:130: warning: ignoring old commands
## /Users/brown/.R/Makevars:132: warning: overriding commands for target `.cc.d'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:132: warning: ignoring old commands
## /Users/brown/.R/Makevars:135: warning: overriding commands for target `.cpp.d'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:135: warning: ignoring old commands
## /Users/brown/.R/Makevars:138: warning: overriding commands for target `.m.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:138: warning: ignoring old commands
```

```
## /Users/brown/.R/Makevars:140: warning: overriding commands for target `.m.d'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:140: warning: ignoring old commands
## /Users/brown/.R/Makevars:143: warning: overriding commands for target `.mm.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:143: warning: ignoring old commands
## /Users/brown/.R/Makevars:145: warning: overriding commands for target `.M.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:145: warning: ignoring old commands
## /Users/brown/.R/Makevars:147: warning: overriding commands for target `.f.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:147: warning: ignoring old commands
## /Users/brown/.R/Makevars:149: warning: overriding commands for target `.f95.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:149: warning: ignoring old commands
## /Users/brown/.R/Makevars:151: warning: overriding commands for target `.f90.o'
## /opt/local/Library/Frameworks/R.framework/Resources/etc/Makeconf:151: warning: ignoring old commands
## SAMPLING FOR MODEL 'model' NOW (CHAIN 1).
##
Iteration:    1 / 10000 [  0%]  (Warmup)
Iteration: 1000 / 10000 [ 10%]  (Warmup)
Iteration: 2000 / 10000 [ 20%]  (Warmup)
Iteration: 3000 / 10000 [ 30%]  (Warmup)
Iteration: 4000 / 10000 [ 40%]  (Warmup)
Iteration: 5000 / 10000 [ 50%]  (Warmup)
Iteration: 6000 / 10000 [ 60%]  (Sampling)
Iteration: 7000 / 10000 [ 70%]  (Sampling)
Iteration: 8000 / 10000 [ 80%]  (Sampling)
Iteration: 9000 / 10000 [ 90%]  (Sampling)
Iteration: 10000 / 10000 [100%]  (Sampling)
## Elapsed Time: 110.502 seconds (Warm-up)
##               83.0415 seconds (Sampling)
##               193.544 seconds (Total)
##
## SAMPLING FOR MODEL 'model' NOW (CHAIN 2).
##
Iteration:    1 / 10000 [  0%]  (Warmup)
Iteration: 1000 / 10000 [ 10%]  (Warmup)
Iteration: 2000 / 10000 [ 20%]  (Warmup)
Iteration: 3000 / 10000 [ 30%]  (Warmup)
Iteration: 4000 / 10000 [ 40%]  (Warmup)
Iteration: 5000 / 10000 [ 50%]  (Warmup)
Iteration: 6000 / 10000 [ 60%]  (Sampling)
Iteration: 7000 / 10000 [ 70%]  (Sampling)
Iteration: 8000 / 10000 [ 80%]  (Sampling)
Iteration: 9000 / 10000 [ 90%]  (Sampling)
Iteration: 10000 / 10000 [100%]  (Sampling)
## Elapsed Time: 104.281 seconds (Warm-up)
##               134.029 seconds (Sampling)
##               238.31 seconds (Total)
##
## SAMPLING FOR MODEL 'model' NOW (CHAIN 3).
##
Iteration:    1 / 10000 [  0%]  (Warmup)
Iteration: 1000 / 10000 [ 10%]  (Warmup)
Iteration: 2000 / 10000 [ 20%]  (Warmup)
Iteration: 3000 / 10000 [ 30%]  (Warmup)
Iteration: 4000 / 10000 [ 40%]  (Warmup)
```

```
Iteration: 5000 / 10000 [ 50%]  (Warmup)
Iteration: 6000 / 10000 [ 60%]  (Sampling)
Iteration: 7000 / 10000 [ 70%]  (Sampling)
Iteration: 8000 / 10000 [ 80%]  (Sampling)
Iteration: 9000 / 10000 [ 90%]  (Sampling)
Iteration: 10000 / 10000 [100%]  (Sampling)
## Elapsed Time: 107.225 seconds (Warm-up)
##               110.478 seconds (Sampling)
##               217.703 seconds (Total)
##
## SAMPLING FOR MODEL 'model' NOW (CHAIN 4).
##
Iteration:    1 / 10000 [  0%]  (Warmup)
Iteration: 1000 / 10000 [ 10%]  (Warmup)
Iteration: 2000 / 10000 [ 20%]  (Warmup)
Iteration: 3000 / 10000 [ 30%]  (Warmup)
Iteration: 4000 / 10000 [ 40%]  (Warmup)
Iteration: 5000 / 10000 [ 50%]  (Warmup)
Iteration: 6000 / 10000 [ 60%]  (Sampling)
Iteration: 7000 / 10000 [ 70%]  (Sampling)
Iteration: 8000 / 10000 [ 80%]  (Sampling)
Iteration: 9000 / 10000 [ 90%]  (Sampling)
Iteration: 10000 / 10000 [100%]  (Sampling)
## Elapsed Time: 97.1059 seconds (Warm-up)
##               46.7754 seconds (Sampling)
##               143.881 seconds (Total)
```