Week 5 assignment: Binomial models

Example solutions

23 February, 2016

About one out of eight women in the U.S. will develop breast cancer at some point in her lifetime. Early diagnoses help with treatment of this potentially fatal disease, and these diagnoses can be made based on a variety of cytological metrics evaluated via biopsy. Your job today is to develop a model that classifies tumors as malignant or benign based on these metrics. The student(s) with the most predictive model will get a prize.

The data are in the `breast_cancer.csv` file. Details for this dataset can be found on the UCI machine learning data repository, which is useful if you ever need data to play with. I split the data into two groups at random: the *training* data, which you'll use to estimate parameters, and the *test* data, which we'll use to evaluate the predictive power of the model. There is a column in the data called `group`, which indicates whether an observation is part of the training or test set.

Data exploration

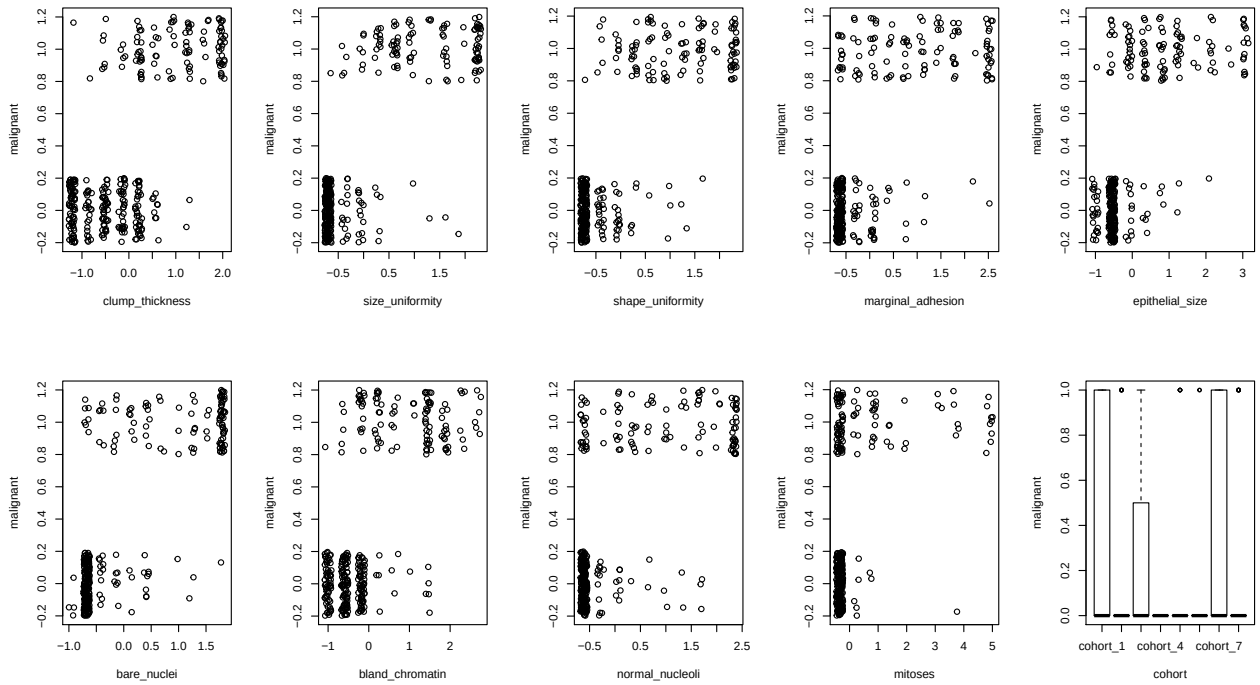
As usual, you will want to explore the data before constructing any statistcal models. Only explore the training data, and do not use the test data for data exploration/visualization. We will pretend that we don't have access to the test data yet.

```
d <- read.csv('data/breast_cancer.csv')
library(dplyr)
str(d)
```

```
## 'data.frame':    699 obs. of  13 variables:
##  $ id               : int  95719 128059 145447 167528 183913 242970 274137 314428 324427 390840 ...
##  $ clump_thickness  : int  6 1 8 4 1 5 8 7 10 8 ...
##  $ size_uniformity  : int  10 1 4 1 2 7 8 9 8 4 ...
##  $ shape_uniformity : int  10 1 4 1 2 7 9 4 8 7 ...
##  $ marginal_adhesion: int  10 1 1 1 1 1 4 10 2 1 ...
##  $ epithelial_size  : int  8 2 2 2 2 5 5 10 3 3 ...
##  $ bare_nuclei      : int  10 5 9 1 1 8 10 3 4 10 ...
##  $ bland_chromatin  : int  7 5 3 3 1 3 7 5 8 3 ...
##  $ normal_nucleoli  : int  10 1 3 6 1 4 8 3 7 9 ...
##  $ mitoses          : int  7 1 1 1 1 1 1 3 8 2 ...
##  $ cohort           : Factor w/ 8 levels "cohort_1","cohort_2",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ group            : Factor w/ 2 levels "test","train": 2 2 2 2 2 2 2 2 2 2 ...
##  $ malignant        : int  1 0 1 0 0 0 1 1 1 1 ...
```

```
# center the continuous explanatory variables
d[, 2:10] <- apply(d[, 2:10], 2, FUN = function(x) unlist(scale(x)))
# create a subsetted data frame with just the training data
d_train <- subset(d, group == 'train')
d_test <- subset(d, group == 'test')
par(mfrow=c(2, 5))
for (i in 2:10){
plot(x = jitter(d_train[, i]),
     y = jitter(d_train[, 'malignant']),
```

```
     xlab = names(d_train)[i], ylab = 'malignant')
}
plot(malignant ~ cohort, data = d_train)
```



Model structure

What is your model? Write it out in LaTeX. Hint: you will want to use a design matrix.

LaTeX here

What is your Stan model statement?

```
data {
  int n; // sample size
  int p; // number of coefficients
  matrix[n, p] X;
  int y[n];
}

parameters {
  vector[p] beta;
}

model {
  beta ~ normal(0, 1);
  y ~ bernoulli_logit(X * beta);
}
```

Building and understanding the design matrix

We mentioned that you would want to use a design matrix. Specifically, your model should be of the form:

$y \sim Bernoulli(p)$

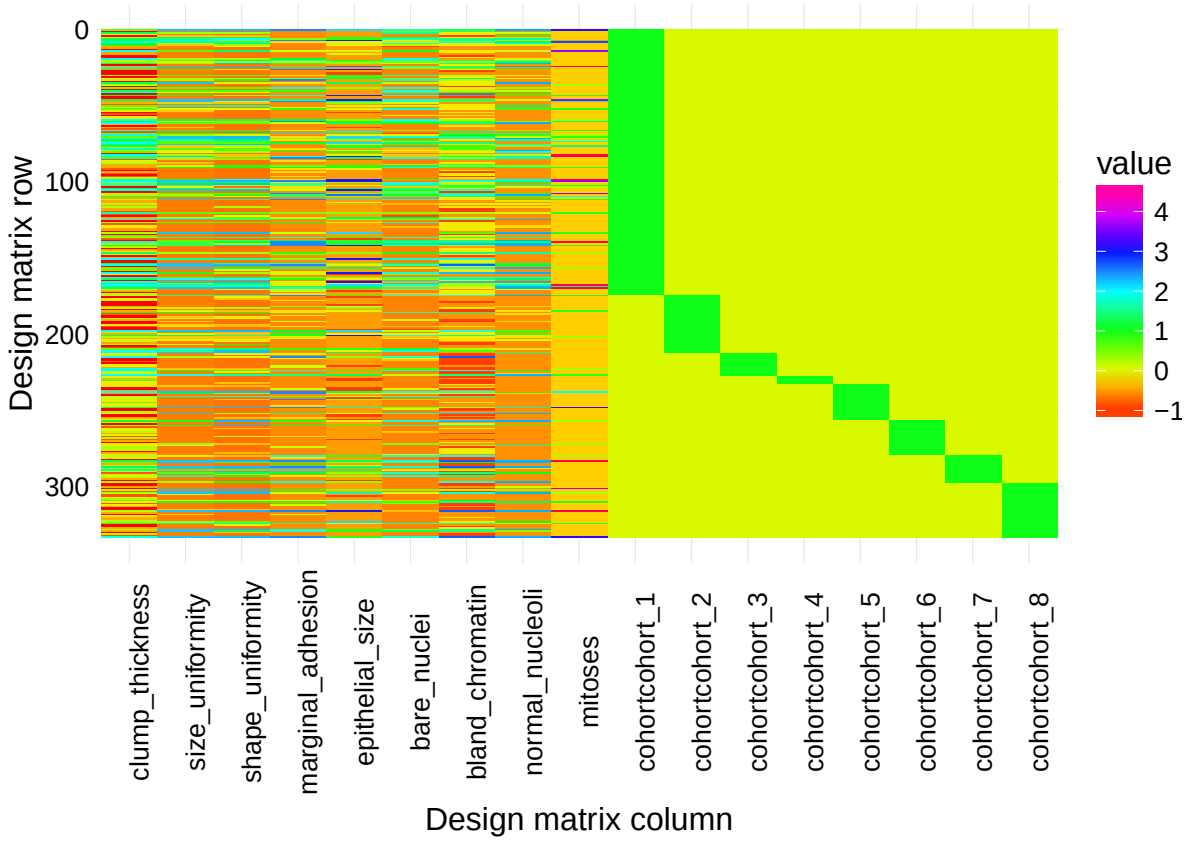
And the probability of malignancy p is modeled using a logit-link:

$log\Big(\frac{p}{1-p}\Big) = X\beta$

The design matrix X contains the tumor features, and also dictates the interpretation of the coefficients β. In the code block below, construct your design matrix, creating an object called `X`. The included code will make an image plot of your design matrix with a horrendous color scheme. Once you fill in your code, set the argument `eval = TRUE` inside of the curly braces at the beginning of the code chuck (this is a chunk option), otherwise the code chunk will not be evaluated when you're knitting your pdf.

```
# define your design matrix below
X <- model.matrix(~ 0 + clump_thickness +
                    size_uniformity +
                    shape_uniformity +
                    marginal_adhesion +
                    epithelial_size +
                    bare_nuclei +
                    bland_chromatin +
                    normal_nucleoli +
                    mitoses +
                    cohort,
                  data = d_train)

# the code below will plot your design matrix
library(reshape2)
library(ggplot2)
mX <- melt(X)
ggplot(mX, aes(x = Var2, y = Var1)) +
  geom_raster(aes(fill = value)) +
  scale_y_reverse() +
  xlab('Design matrix column') +
  ylab('Design matrix row') +
  scale_fill_gradientn(colors = rainbow(20)) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle=90))
```



For each column of X you will get a coefficient, one element in β. For instance, the coefficient β_1 will be associated with the first column in X, which we might denote $X[, 1]$, to borrow some R syntax. There's no sense in estimating parameters if you don't know what they mean (Abraham Lincoln said that), so below, list each element in β and briefly describe what it represents/how you would interpret it:

1. β_1 represents the increase in the logit probability of malignance for an increase of one standard deviation in clump thickness
2. β_2 represents the increase in the logit probability of malignance for an increase of one standard deviation in size uniformity
3. β_3 represents the increase in the logit probability of malignance for an increase of one standard deviation in shape uniformity
4. β_4 represents the increase in the logit probability of malignance for an increase of one standard deviation in marginal adhesion
5. β_5 represents the increase in the logit probability of malignance for an increase of one standard deviation in epithelial size
6. β_6 represents the increase in the logit probability of malignance for an increase of one standard deviation in bare nuclei
7. β_7 represents the increase in the logit probability of malignance for an increase of one standard deviation in bland chromatin
8. β_8 represents the increase in the logit probability of malignance for an increase of one standard deviation in normal nucleoli
9. β_9 represents the increase in the logit probability of malignance for an increase of one standard deviation in mitoses

The remaining columns (10 through 17) are group-level intercepts, whose coefficients will represent the logit probability of malignance for an average tumor.

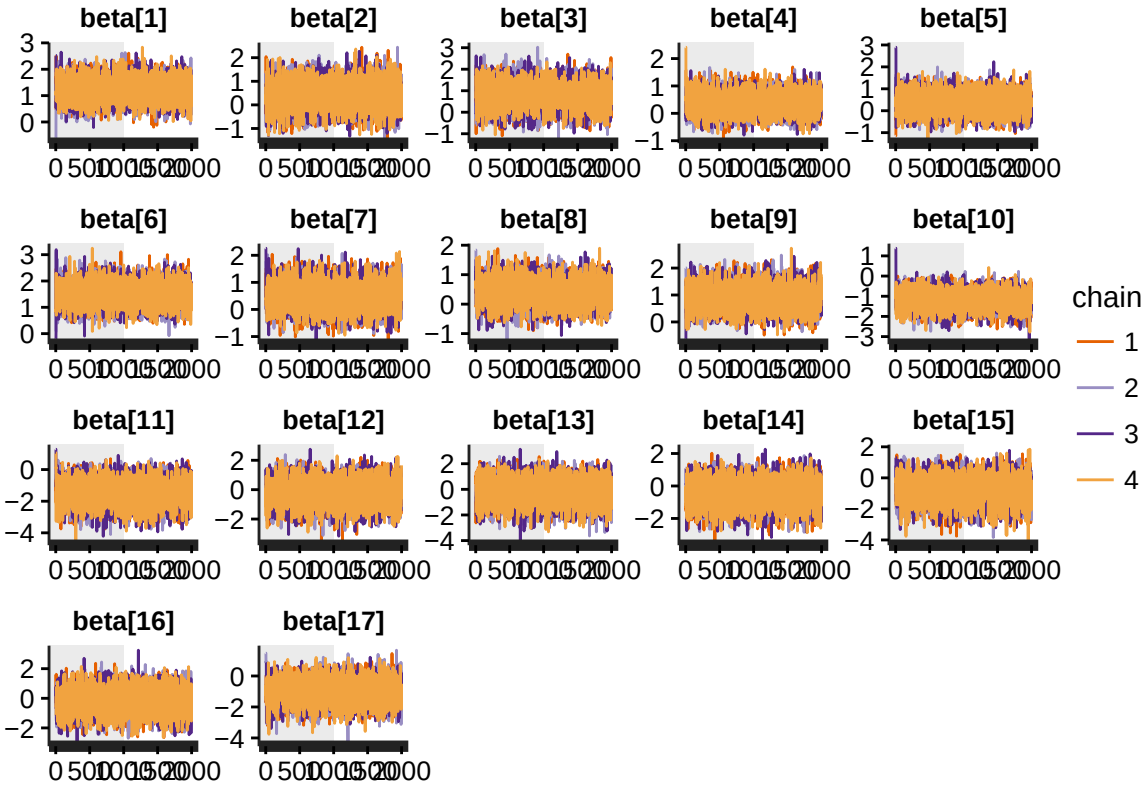
Parameter estimation

Use the **training** data to estimate your model's parameters (`group == 'test'`). Do not use the **test** data yet. Make sure that the MCMC algorithm has converged before moving forward.

```
library(rstan)
rstan_options(auto_write = TRUE)
stan_d <- list(n = nrow(X),
               p = ncol(X),
               X = X,
               y = d_train$malignant)
m_fit <- stan('bernoulli_glm.stan',
              data = stan_d, cores = 2)
m_fit
```

```
## Inference for Stan model: bernoulli_glm.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##            mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## beta[1]    1.20    0.01 0.41   0.43   0.92   1.19   1.47   2.04  3228    1
## beta[2]    0.39    0.01 0.56  -0.67   0.01   0.38   0.76   1.53  2863    1
## beta[3]    0.78    0.01 0.54  -0.31   0.42   0.80   1.14   1.85  2478    1
## beta[4]    0.39    0.01 0.36  -0.30   0.13   0.39   0.63   1.10  2719    1
## beta[5]    0.36    0.01 0.44  -0.51   0.06   0.36   0.66   1.23  3122    1
## beta[6]    1.49    0.01 0.41   0.72   1.21   1.48   1.76   2.32  2977    1
## beta[7]    0.50    0.01 0.47  -0.42   0.18   0.49   0.80   1.40  3271    1
## beta[8]    0.40    0.01 0.39  -0.37   0.14   0.41   0.66   1.15  3406    1
## beta[9]    0.90    0.01 0.44   0.08   0.60   0.89   1.21   1.78  2902    1
## beta[10]  -1.18    0.01 0.44  -2.04  -1.48  -1.18  -0.88  -0.34  3143    1
## beta[11]  -1.48    0.01 0.73  -2.95  -1.98  -1.46  -0.98  -0.08  3009    1
## beta[12]  -0.20    0.01 0.83  -1.89  -0.75  -0.18   0.37   1.39  3561    1
## beta[13]  -0.22    0.02 0.92  -2.10  -0.84  -0.21   0.41   1.51  3636    1
## beta[14]  -0.46    0.01 0.77  -2.03  -0.97  -0.45   0.08   0.98  3128    1
## beta[15]  -0.81    0.01 0.80  -2.47  -1.33  -0.79  -0.25   0.66  3679    1
## beta[16]  -0.16    0.01 0.79  -1.76  -0.70  -0.16   0.37   1.39  3682    1
## beta[17]  -1.00    0.01 0.73  -2.48  -1.47  -0.98  -0.50   0.40  3687    1
## lp__     -47.68    0.08 3.00 -54.46 -49.47 -47.29 -45.52 -42.89  1559    1
##
## Samples were drawn using NUTS(diag_e) at Tue Feb 23 09:14:01 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
traceplot(m_fit, inc_warmup = TRUE, 'beta')
```



Out of sample predictive power

One measure of a model's ability to predict new data is the log likelihood of new data, given the parameters of the model $[\tilde{y} \mid \theta]$, where $\tilde{y}$ is the new data (the **test** or **validation** data), and the parameters θ have been estimated from other data (e.g., the **training** data).

Hints:

- this is done most easily via a new design matrix X_{test}, which can be multiplied by the vector of model parameters, and must be declared in the `data` block
- make sure that if you used any feature scaling or centering in the training data, that the exact same scaling/centering schemes are applied to the test set
- you'll use the `generated quantities` block to calculate the log-likelihood of the test data
- you can obtain the joint log likelihood with the `bernoulli_logit_log` function in Stan, and I wrote a generated quantities model block for you below, which should be the last block in your new Stan model statement

What is your updated Stan model?

```
data {
  int n; // sample size
  int p; // number of coefficients
  matrix[n, p] X;
  int y[n];
  int n_test;
  int y_test[n_test];
```

```
  matrix[n_test, p] X_test;
}

parameters {
  vector[p] beta;
}

model {
  beta ~ normal(0, 1);
  y ~ bernoulli_logit(X * beta);
}


generated quantities {
  // I wrote this section for you as a hint
  real loglik_test;
  vector[n_test] logit_p_test;

  logit_p_test <- X_test * beta;
  loglik_test <- bernoulli_logit_log(y_test, logit_p_test);
  //returns the sum of the log likelihoods (the joint log-likelihood)
}
```

Acquire the posterior distribution of the model parameters and the holdout log likelihood.

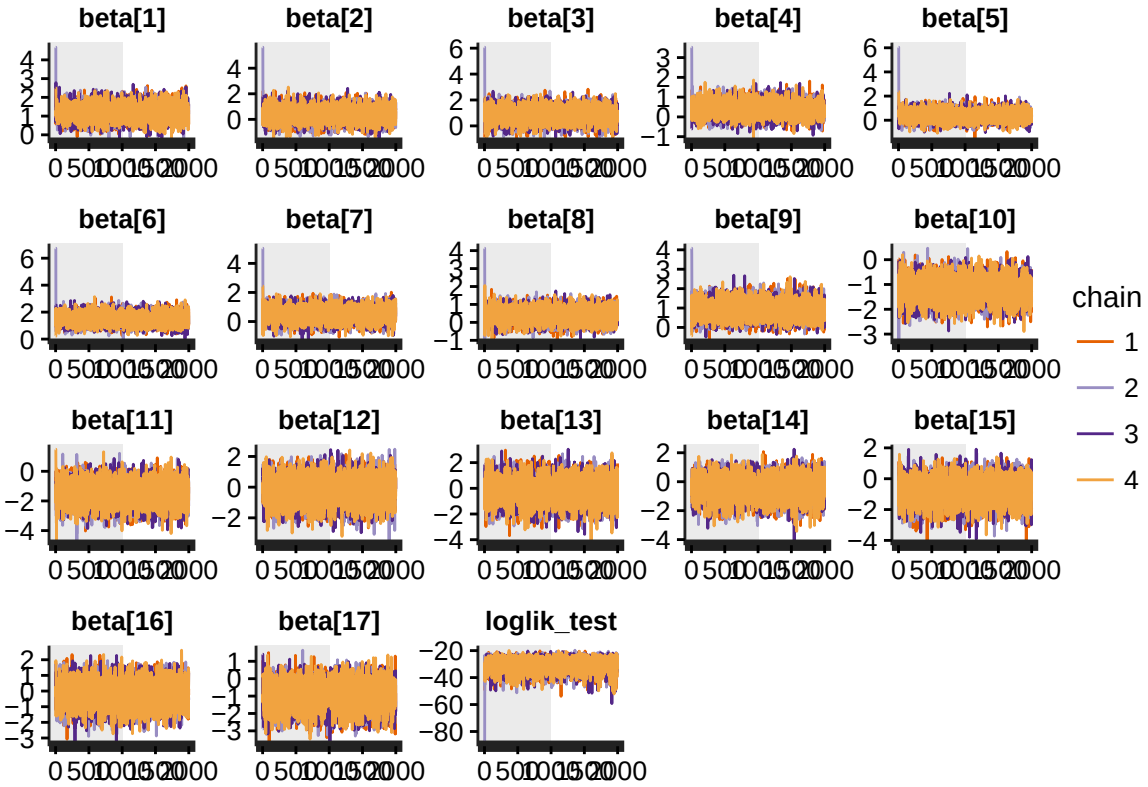
```
X_test <-  model.matrix(~ 0 + clump_thickness +
                     size_uniformity +
                     shape_uniformity +
                     marginal_adhesion +
                     epithelial_size +
                     bare_nuclei +
                     bland_chromatin +
                     normal_nucleoli +
                     mitoses +
                     cohort,
                   data = d_test)

stan_d <- list(n = nrow(X),
               p = ncol(X),
               X = X,
               y = d_train$malignant,
               n_test = nrow(X_test),
               y_test = d_test$malignant,
               X_test = X_test)
m_fit <- stan('bernoulli_glm_test.stan',
              data = stan_d, cores = 2)
print(m_fit, pars = c('beta', 'loglik_test'))
```

```
## Inference for Stan model: bernoulli_glm_test.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##               mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff
```

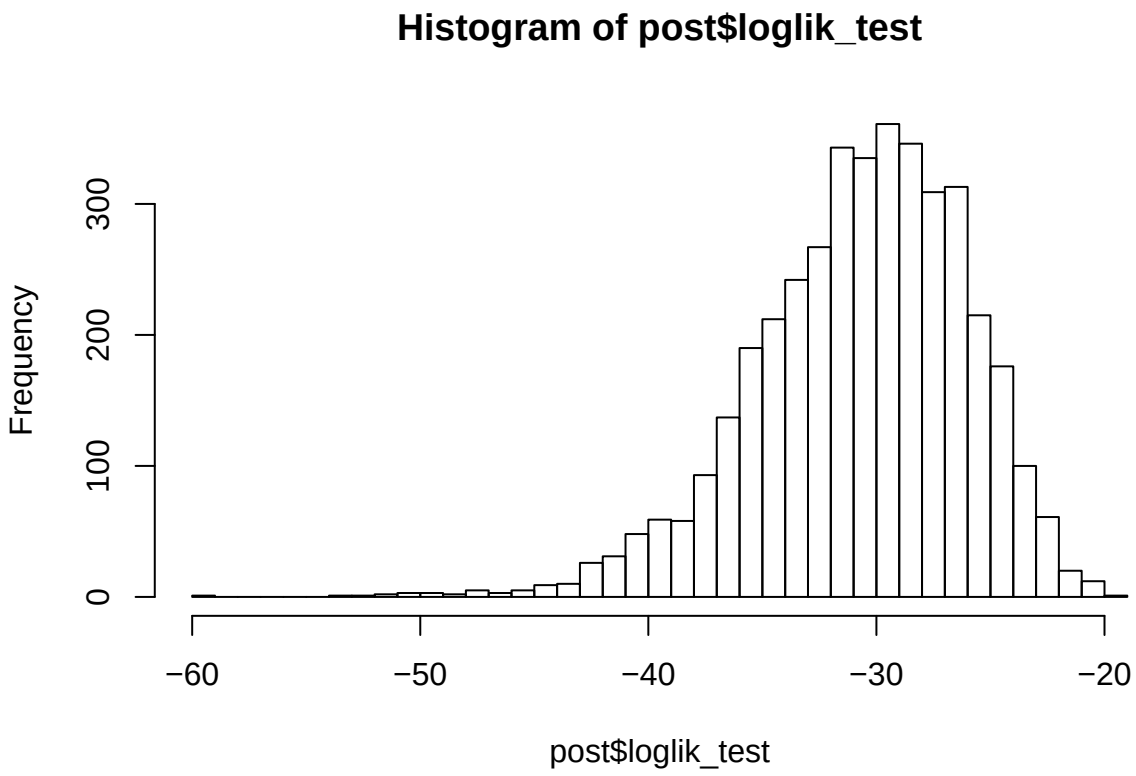
```
## beta[1]       1.19    0.01 0.41   0.41   0.91   1.17   1.45   2.06  2790
## beta[2]       0.37    0.01 0.56  -0.66  -0.02   0.37   0.73   1.50  2220
## beta[3]       0.81    0.01 0.56  -0.23   0.41   0.80   1.19   1.91  3075
## beta[4]       0.39    0.01 0.36  -0.31   0.15   0.39   0.63   1.13  3148
## beta[5]       0.38    0.01 0.43  -0.44   0.08   0.38   0.68   1.23  3311
## beta[6]       1.48    0.01 0.41   0.68   1.20   1.48   1.75   2.32  4000
## beta[7]       0.49    0.01 0.47  -0.43   0.18   0.49   0.81   1.41  3261
## beta[8]       0.39    0.01 0.39  -0.38   0.12   0.39   0.65   1.14  2816
## beta[9]       0.90    0.01 0.44   0.07   0.60   0.90   1.18   1.77  4000
## beta[10]     -1.18    0.01 0.44  -2.05  -1.46  -1.17  -0.88  -0.36  3545
## beta[11]     -1.46    0.01 0.75  -2.98  -1.95  -1.45  -0.95  -0.05  4000
## beta[12]     -0.22    0.01 0.84  -1.89  -0.78  -0.20   0.36   1.38  4000
## beta[13]     -0.21    0.01 0.93  -2.06  -0.84  -0.19   0.43   1.55  4000
## beta[14]     -0.46    0.01 0.78  -2.04  -0.99  -0.43   0.07   0.98  3627
## beta[15]     -0.82    0.01 0.79  -2.42  -1.35  -0.79  -0.27   0.65  3762
## beta[16]     -0.17    0.01 0.78  -1.73  -0.69  -0.16   0.36   1.32  4000
## beta[17]     -1.00    0.01 0.74  -2.49  -1.49  -1.01  -0.49   0.43  4000
## loglik_test -30.72    0.08 4.67 -41.06 -33.59 -30.24 -27.33 -23.10  3299
##             Rhat
## beta[1]        1
## beta[2]        1
## beta[3]        1
## beta[4]        1
## beta[5]        1
## beta[6]        1
## beta[7]        1
## beta[8]        1
## beta[9]        1
## beta[10]       1
## beta[11]       1
## beta[12]       1
## beta[13]       1
## beta[14]       1
## beta[15]       1
## beta[16]       1
## beta[17]       1
## loglik_test    1
##
## Samples were drawn using NUTS(diag_e) at Tue Feb 23 09:14:42 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
traceplot(m_fit, inc_warmup = TRUE, c('beta', 'loglik_test'))
```



Make a histogram of the holdout log likelihood and report the posterior mean along with a 95% credible interval.

```
post <- rstan::extract(m_fit)
par(mfrow=c(1, 1))
hist(post$loglik_test, breaks=40)
```




```
c(mean = mean(post$loglik_test), quantile(post$loglik_test, c(0.025, 0.975)))
```

```
##      mean      2.5%     97.5%
## -30.72121 -41.05747 -23.09541
```

Showing predictions

The whole point of building this model is to diagnose whether a tumor is malignant based on some features. Plot the posterior probability of tumor malignance for each holdout tumor, and show the true tumor status in the same graph. Multiple graph types are possible here, but we do not recommend simply copying and pasting code from another example (so far about a quarter of plots made in this way have made sense). Instead, think hard about what sort of data display would be effective, and make that plot!

```
library(reshape2)
p_df <- melt(post$logit_p_test, varnames = c('iter', 'obs'))

subset(d, group == 'test') %>%
  mutate(obs = 1:n()) %>%
  full_join(p_df) %>%
  ggplot(aes(x = value,
             group = obs,
             fill = factor(malignant))) +
  geom_density(alpha = .1) +
  facet_wrap(~ cohort, nrow = 2) +
  theme_bw() +
  xlab('Predicted logit probability of tumor malignance') +
```

```
  ylab('Posterior density') +
  theme(legend.position = "top")
```

```
## Joining by: "obs"
```