Bayesian Regression Analysis

Joseph Parks

Introduction

I was given a dataset that had one response, Y, and three predictors, var1, var2, and var3. Otherwise, I was told nothing else about the dataset. So, I began with exploratory data analysis. After importing the data into R, I printed out the raw data to look at, as well as, a summary of the data.

```
> df = read.csv("/Users/parksjg/Desktop/Stats/Bayes/project/ProjectData.csv", header = TRUE, sep = "\t")
> summary(df)
       Y              var1            var2              var3         
 Min.   : 9.513   Min.   :2.661   Min.   :-2.24149   Min.   :-3.49347  
 1st Qu.:16.196   1st Qu.:4.199   1st Qu.:-0.69344   1st Qu.:-1.01172  
 Median :19.929   Median :4.897   Median :-0.09727   Median :-0.04457  
 Mean   :20.646   Mean   :4.895   Mean   :-0.06782   Mean   : 0.01622  
 3rd Qu.:25.840   3rd Qu.:5.487   3rd Qu.: 0.50123   3rd Qu.: 0.83447  
 Max.   :34.669   Max.   :7.214   Max.   : 3.11944   Max.   : 3.64452  
```

I obtained useful statistics like the sample median and sample mean for all the variables. I was also able to calculate the standard deviations of each variable, as shown below.

$$\sigma_Y = 5.775877$$

$$\sigma_{var1} = 0.9755867$$

$$\sigma_{var2} = 1.013836$$

$$\sigma_{var3} = 1.433017$$

Then I performed linear regression on the data with Y as the response and var1, var2, and var3 as the predictors.

```
> lmod = lm(df$Y ~ ., data = df)
> summary(lmod)

Call:
lm(formula = df$Y ~ ., data = df)

Residuals:
     Min       1Q   Median       3Q      Max 
-11.0651  -4.3088   0.2126   4.2177  11.3239 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  10.2960     2.5340   4.063 8.93e-05 ***
var1          2.1369     0.5083   4.204 5.24e-05 ***
var2          1.3093     0.6903   1.897  0.06041 .  
var3         -1.3668     0.4905  -2.787  0.00624 ** 
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.331 on 114 degrees of freedom
Multiple R-squared:  0.1698,	Adjusted R-squared:  0.148 
F-statistic: 7.772 on 3 and 114 DF,  p-value: 9.095e-05
```

The p-value for var2 is slightly high and would need further investigation. The adjusted R-squared shows that this linear model explains 14.8% of the variability observed in the data, which is very low. Also, the variance inflation factors show collinearity again between var2 and var3.

```
> vif(lmod)
    var1     var2     var3 
1.012144 2.016255 2.033481 
```

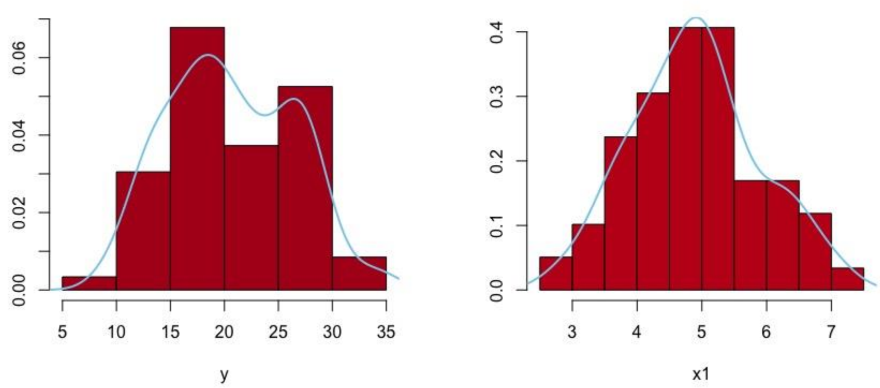
Next, I performed linear regression again, but only with var1 and var3.

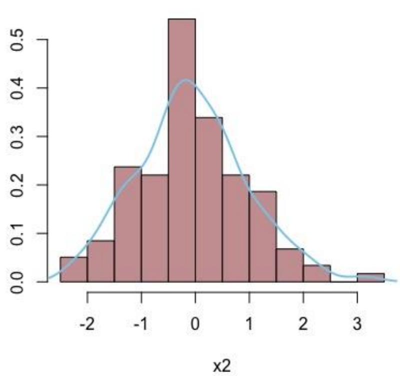
```
> lmod2 = lm(df$Y ~ df$var1 + df$var3, data = df)
> summary(lmod2)

Call:
lm(formula = df$Y ~ df$var1 + df$var3, data = df)

Residuals:
     Min       1Q   Median       3Q      Max 
-10.2886  -4.2850   0.4284   4.3220  12.4803 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  10.3917     2.5620   4.056 9.12e-05 ***
df$var1       2.0970     0.5136   4.083 8.24e-05 ***
df$var3      -0.7069     0.3496  -2.022   0.0455 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.391 on 115 degrees of freedom
Multiple R-squared:  0.1436,	Adjusted R-squared:  0.1287 
F-statistic: 9.642 on 2 and 115 DF,  p-value: 0.0001345
```

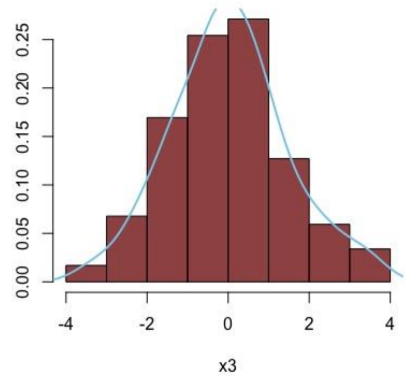
Now, each of the predictors are significant at the alpha = 5% level. But the adjusted R-squared is worse than the full model with var1, var2, and var3. The linear equation obtained from the full linear regression is below.

$$Y_i = \beta_0 + \beta_1 \mathrm{var1}_i + \beta_2 \mathrm{var2}_i + \beta_3 \mathrm{var3}_i \implies Y_i = (10.2960) + (2.1369)\mathrm{var1}_i + (1.3093)\mathrm{var2}_i + (-1.3668)\mathrm{var3}_i$$

Below are histograms of all four variables, and the blue line is the estimated densities. Note that Y is bimodal. Also, x1 (var1), x2 (var2), and x3 (var3) are unimodal but don’t appear very normal.







2 Component Mixture

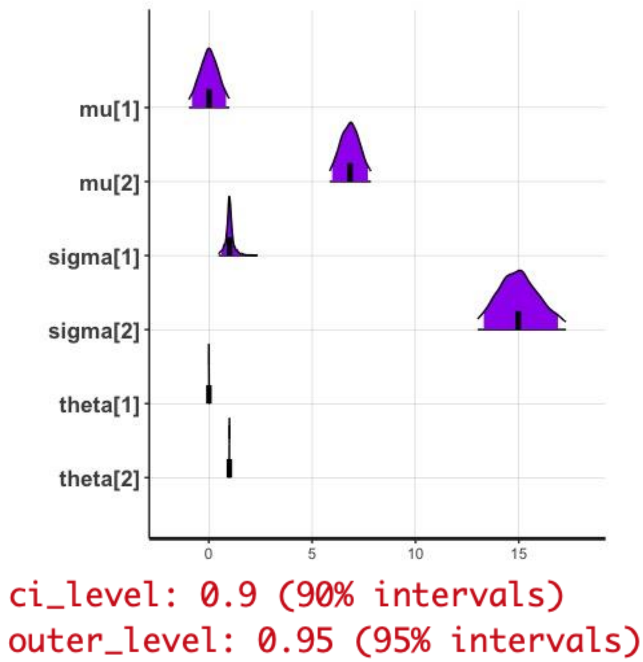
First, I tried a two-component mixture model for estimating the means for var1 and var3. I used a Cauchy prior on sigma and Normal priors on the means.

```
model {
  vector[K] log_theta = log(theta);
  sigma ~ cauchy(1,0.1);
  mu[1] ~ normal(0,0.5);
  mu[2] ~ normal(5,0.5);
  for (n in 1:N) {
    vector[K] lps = log_theta;
    for (k in 1:K)
      lps[k] +=normal_lpdf(y[n] | mu[k], sigma[k]);
    target += log_sum_exp(lps);
  }
}
```

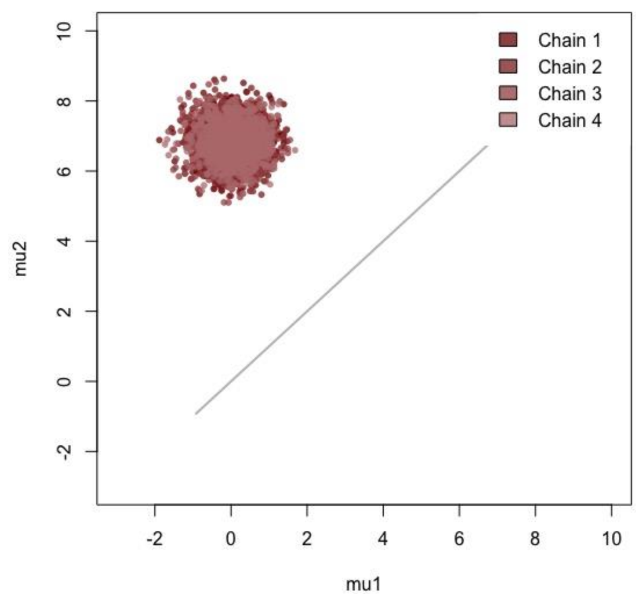
The means were ordered in the stan model, so mu[1] is the mean of var3 and mu[2] is the mean of var1. Notice that mu[1] and mu[2] are close to the sample means calculated above. All four chains converged with no divergences and the Rhat values around 1 also show convergence.

```
> print(degenerate_fit)
Inference for Stan model: 79f7b61d9d8eda9dd608879f1417d10b.
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
mu[1]       0.01    0.01 0.50   -0.97   -0.31    0.01    0.34    0.99  1307 1.00
mu[2]       6.83    0.01 0.52    5.85    6.48    6.83    7.18    7.84  2853 1.00
sigma[1]    1.17    0.07 1.65    0.47    0.91    1.00    1.10    2.36   519 1.01
sigma[2]   15.01    0.02 1.09   13.01   14.27   14.97   15.72   17.28  2130 1.00
theta[1]    0.01    0.00 0.01    0.00    0.00    0.01    0.01    0.03  3511 1.00
theta[2]    0.99    0.00 0.01    0.97    0.99    0.99    1.00    1.00  3511 1.00
lp__     -507.68    0.07 2.00 -512.39 -508.76 -507.31 -506.17 -504.97   891 1.01
```

Sigma[2] is large and I'm not sure why. At one point it was small but I changed the stan model so much that I don't remember how I got the better estimate of sigma[2].



Also, the values of theta[1] and theta[2] sum to one. However, since theta[2] = 0.99, that means we sample from the second component 99% of the time. It seems that the mode of mu[2] is the strongest. As seen below, all four chains favor the mode of var1, which again is mu[2].



3 Component Mixture

Next, I tried a three-component mixture model with normal priors on the means and standard deviations. In addition to the three-component mixture, the stan model below also performs Bayesian linear regression for the full model.

```
model {
  vector[K] log_theta = log(theta);
  sigma[1] ~ normal(0,1);
  sigma[2] ~ normal(0,1);
  sigma[3] ~ normal(0,1);
  mu[1] ~ normal(-0.06782,0.1);
  mu[2] ~ normal(0.01622,0.1);
  mu[3] ~ normal(4.895,0.1);
  beta ~ normal(0, 5) ;       // prior for betas
  tao ~ cauchy(1, 0.1) ;    // prior for sigma
  y ~ normal(X*beta, tao) ; // vectorized likelihood
  for (n in 1:N) {
    vector[K] lps = log_theta;
    for (k in 1:K)
      lps[k] +=normal_lpdf(y[n] | mu[k], sigma[k]);
      //lps[k] +=lognormal_lpdf(y[n] | mu[k], sigma[k]);
      //lps[k] +=student_t_lpdf(y[n] | 117, mu[k], sigma[k]);
    target += log_sum_exp(lps);
  }
}

generated quantities{
  vector[N] y_rep ; // vector of same length as the data y
  vector[N] y_rep_reg ; // vector of same length as the data y
  vector[N] y_rep_normal;
  vector[N] y_rep_err;
  vector[N] y_gam;
 {
  for (n in 1:N) {
    vector[K] lps = theta;
    for (k in 1:K)
      lps[k] +=normal_rng(X[n]*beta, tao);
    y_rep[n] = log_sum_exp(lps);
  }

  for (n in 1:N) {
    y_rep_normal[n] = normal_rng(X[n]*beta, tao);
    y_rep_reg[n] = X[n]*beta;
    y_rep_err[n] = X[n]*beta + normal_rng(0, tao);
    y_gam[n] = gamma_rng(20,1);
  }
 }
}
'
```

Note that X is a matrix with the first column of ones, the second column is var1, the third column is var2, and the fourth column is var3.

```
X <- cbind(Const = 1, X1 = x1, X2 = x2, X3 = x3)
J <- ncol(X)

# Prepare the data we'll need as a list
stan_data <- list(y = y, N = N, J = J, K = K, X = X)

stan_code <- '
data {
int<lower = 1> N;  // integer, number of observations
int<lower = 1> J;  // integer, number of columns in model matrix
int<lower = 1> K;
matrix[N,J]   X ; // N by J model matrix
vector[N] y;
}

parameters {
vector<lower=0>[K] sigma;
```

```
ordered[K] mu;
real<lower=0> tao ; // real number > 0, standard deviation
vector[J]     beta ;  // J-vector of regression coefficients
simplex[K] theta;
}
```

Also, beta is a vector of length four. Here again the means are ordered, and tao is the estimated standard deviation of the response Y.

```
> print(degenerate_fit)
Inference for Stan model: b00a7cb654e3367d33176728eddf0a6b.
4 chains, each with iter=10000; warmup=5000; thin=1;
post-warmup draws per chain=5000, total post-warmup draws=20000.

             mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
sigma[1]     1.46    0.12 9.44   0.42   0.92   1.01   1.11   3.40  6632    1
sigma[2]     1.35    0.07 3.17   0.45   0.92   1.01   1.11   3.25  2320    1
sigma[3]    16.72    0.01 1.09  14.75  15.97  16.66  17.41  19.05 22939    1
mu[1]       -0.10    0.00 0.09  -0.28  -0.16  -0.10  -0.04   0.06 15096    1
mu[2]        0.05    0.00 0.09  -0.11  -0.01   0.05   0.11   0.22 16991    1
mu[3]        4.96    0.00 0.10   4.77   4.89   4.96   5.03   5.16 21760    1
tao          5.34    0.00 0.35   4.70   5.09   5.32   5.57   6.08 20291    1
beta[1]      8.28    0.02 2.25   3.88   6.79   8.31   9.79  12.72 10267    1
beta[2]      2.53    0.00 0.45   1.64   2.23   2.53   2.83   3.42 10268    1
beta[3]      1.28    0.01 0.68  -0.04   0.82   1.28   1.74   2.63 14433    1
beta[4]     -1.37    0.00 0.49  -2.33  -1.70  -1.37  -1.05  -0.42 14250    1
theta[1]     0.01    0.00 0.01   0.00   0.00   0.01   0.01   0.03 16917    1
theta[2]     0.01    0.00 0.01   0.00   0.00   0.01   0.01   0.03 15217    1
theta[3]     0.98    0.00 0.01   0.95   0.98   0.99   0.99   1.00 17304    1
```

All four chains converged, and the estimated means are close to the sample means. And tao is close to the sample standard deviation of the response Y.

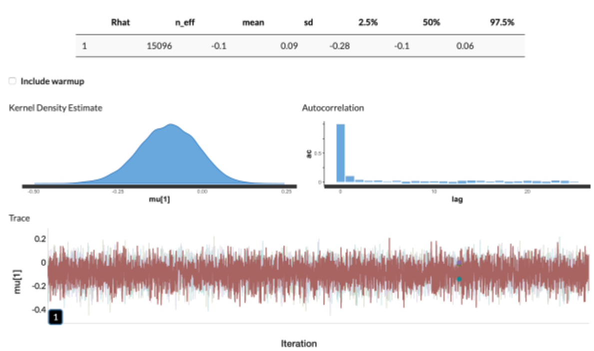
mu[3] = 4.96 ~ Sample mean(var1) = 4.895306

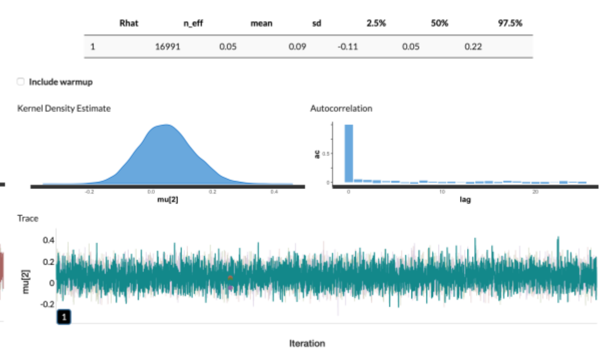
mu[2] = 0.05 ~ Sample mean(var3) = 0.01622395

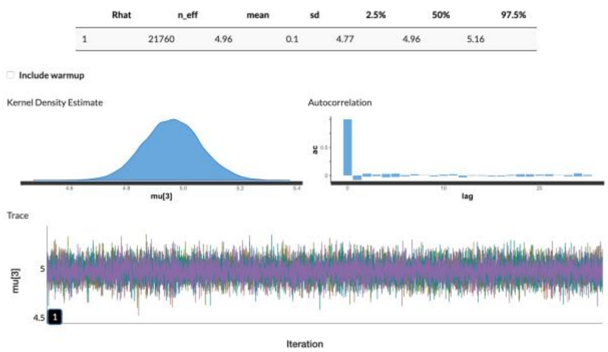
mu[1] = -0.10 ~ Sample mean(var2) = -0.06782252

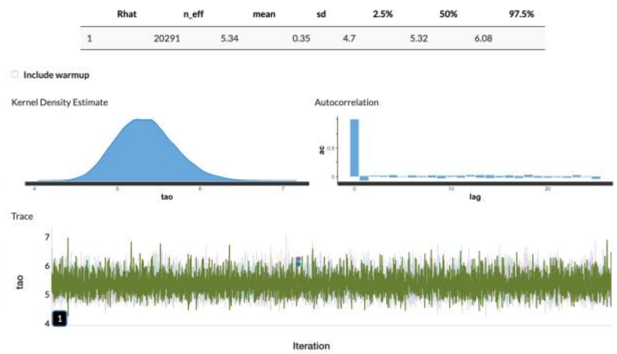
tau = 5.34 ~ Sample standard deviation(Y) = 5.775877

Below are the kernel density estimates, autocorrelation, and trace plots for mu[1], mu[2], mu[3], and tao.









Also, the betas are the coefficients for the Bayesian linear equation which is given below. Notice the Bayesian linear equation is close to the linear equation obtained from the linear regression.

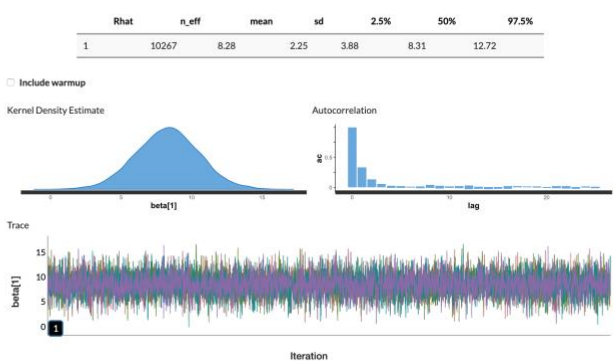
Linear Equation

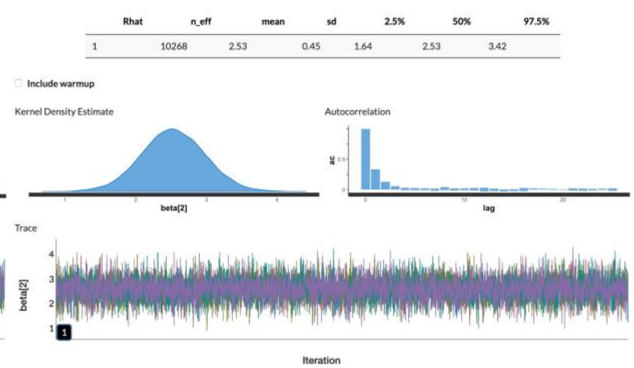
$$Y_i = (10.2960) + (2.1369)\text{var1}_i + (1.3093)\text{var2}_i + (-1.3668)\text{var3}_i$$

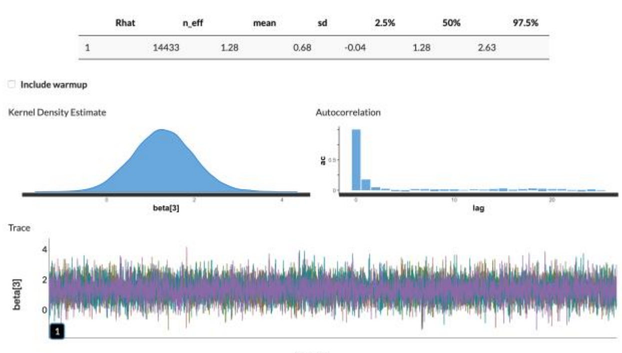
Bayesian Linear Equation

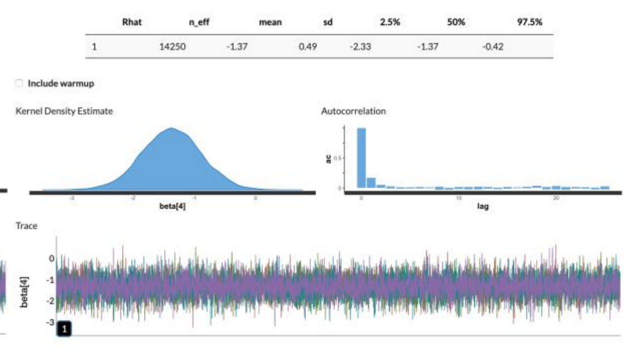
$$Y_i = 8.28 + (2.53)\text{var1}_i + (1.28)\text{var2}_i + (-1.37)\text{var3}_i$$

Below are the kernel density estimates, autocorrelation, and trace plots for all four beta coefficients.

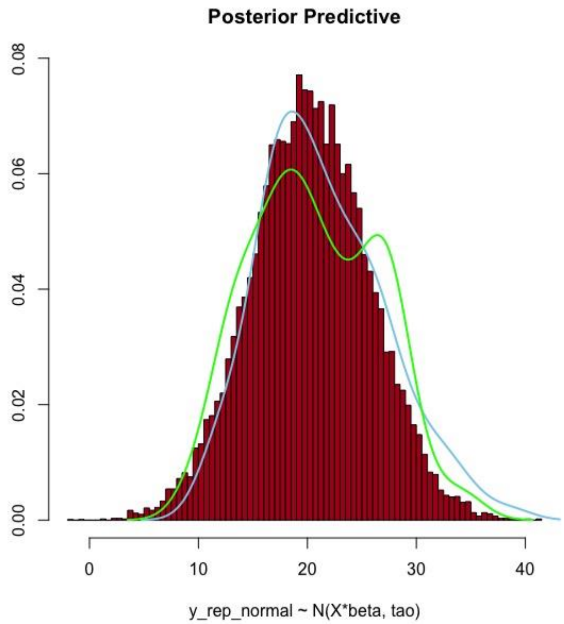






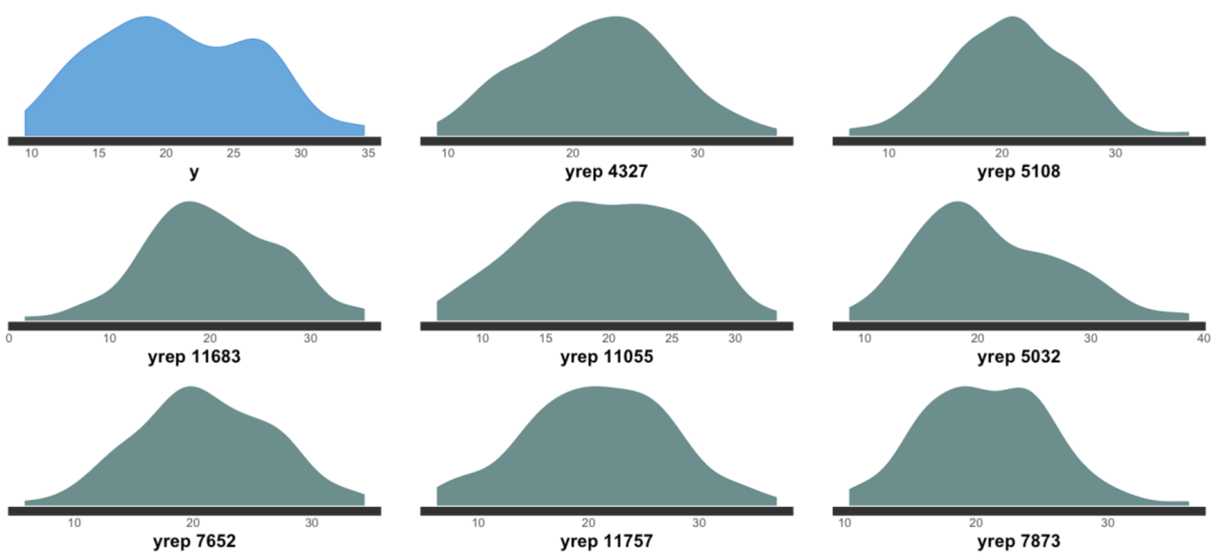


Using the Bayesian linear equation, I wanted to test the posterior predictive distribution, so in the generated quantities block of the stan model I generate new Y's called y_rep_normal. Next I compare my results to the true observations Y.

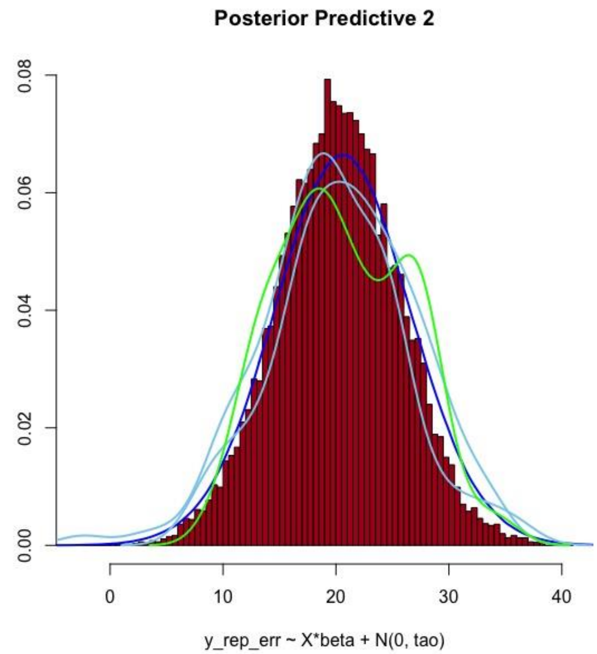


The green line is the density of the true observations Y, the histogram and the blue line are samples from the posterior predictive distributions. The posterior distribution is having a hard time replicating the bimodal nature of the observations Y.

Below is the distribution of observed data, Y, and random samples from the posterior predictive, y_rep_normal. As we can see some of the samples capture the bimodality a little bit, but some clearly do not.



I also generated another posterior predictive called y_rep_err that was modeled as X*beta + N(0, tao). And to little surprise it behaved very similarly to y_rep_normal.



Again, the green line is the density of Y, the true observations that I am trying to model. The dark blue line is the average of my MCMC samples, while the red histogram and the light blue lines are the estimated posterior predictive densities from random samples.

The MSE for y_rep_normal is 69.8634, and the MSE for y_rep_err is 69.92152.

Conclusion

The three-component mixture does better than the two-component mixture, despite the collinearity between var2 and var3. In the end, I would keep all three predictors in the model. I would also like to spend more time working with different priors as the normal priors don't seem to fit very well. I think I need something wider like a gamma or beta. I would also like to perform principal component regression to reduce the collinearity and decrease the variance inflation factors. I am also interested in seeing how a generalized additive model might fit the data.