Lab 9

Bingying Liu

3/30/2020

1. Linear Regression

Frequentist Approach

```
data("clouds", package = "HSAUR3")
#head(clouds)

# frequentist approach
ols <- lm(rainfall ~ seeding * (sne + cloudcover + prewetness + echomotion) + time,
          data = clouds)
summary(ols)
```

```
##
## Call:
## lm(formula = rainfall ~ seeding * (sne + cloudcover + prewetness +
##     echomotion) + time, data = clouds)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5259 -1.1486 -0.2704  1.0401  4.3913
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -0.34624    2.78773  -0.124  0.90306
## seedingyes                     15.68293    4.44627   3.527  0.00372 **
## sne                             0.41981    0.84453   0.497  0.62742
## cloudcover                      0.38786    0.21786   1.780  0.09839 .
## prewetness                      4.10834    3.60101   1.141  0.27450
## echomotionstationary            3.15281    1.93253   1.631  0.12677
## time                           -0.04497    0.02505  -1.795  0.09590 .
## seedingyes:sne                 -3.19719    1.26707  -2.523  0.02545 *
## seedingyes:cloudcover          -0.48625    0.24106  -2.017  0.06482 .
## seedingyes:prewetness          -2.55707    4.48090  -0.571  0.57796
## seedingyes:echomotionstationary -0.56222    2.64430  -0.213  0.83492
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.205 on 13 degrees of freedom
## Multiple R-squared:  0.7158, Adjusted R-squared:  0.4972
## F-statistic: 3.274 on 10 and 13 DF,  p-value: 0.02431
```

Exercise 1: Interpret the significant coefficients (at the 0.05 significance level).

- Using 0.05 as significance level, we can see that “seedingyes” and “seedinghes:sne” are statistically significant.
- seedingyes: when sne, cloudcover, prewetness and echomotion are zero, the change of seeding from no to yes has will increase the amount of rainfall by 15.68.
- seedingyes:sne : Holding all other variables the same, for cloud with no seeding, the effect of one unit of increase in sne is 0.4198; for cloud with seeding, the effect of one unit of increase in sne is 0.4198 - 3.1953 = -2.7755

Bayesian Approach

```
beta0.prior <- cauchy()
beta.prior <- cauchy()

stan.glm <- stan_glm(data = clouds,
                     formula = rainfall ~ seeding * (sne + cloudcover + prewetness + echomotion) + time,
                     family = gaussian(),
                     prior = beta.prior,
                     prior_intercept = beta0.prior,
                     refresh = 0,
                     refresh = 0)
summary(stan.glm)
```

```
## 
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      rainfall ~ seeding * (sne + cloudcover + prewetness + echomotion) + 
##     time
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 24
##  predictors:   11
## 
## Estimates:
##                                    mean   sd   10%   50%   90%
## (Intercept)                       1.4    2.9 -2.2   1.3   5.0 
## seedingyes                       11.7    4.7  5.6  11.9  17.7 
## sne                               0.0    0.9 -1.1   0.0   1.1 
## cloudcover                        0.3    0.2  0.0   0.3   0.6 
## prewetness                        3.9    3.6 -0.6   3.9   8.6 
## echomotionstationary              2.7    1.9  0.3   2.7   5.0 
## time                              0.0    0.0 -0.1   0.0   0.0 
## seedingyes:sne                   -2.2    1.3 -3.9  -2.2  -0.5 
## seedingyes:cloudcover            -0.4    0.2 -0.7  -0.4  -0.1 
## seedingyes:prewetness            -2.8    4.4 -8.4  -2.8   2.9 
## seedingyes:echomotionstationary -0.2    2.5 -3.4  -0.2   3.1 
## sigma                             2.4    0.5  1.8   2.3   3.0 
## 
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.4    0.7  3.6   4.4   5.3  
## 
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for details see help('summa
ry.stanreg')).
## 
## MCMC diagnostics
##                                 mcse Rhat n_eff
## (Intercept)                     0.1  1.0  2353 
## seedingyes                      0.1  1.0  1774 
## sne                             0.0  1.0  2015 
## cloudcover                      0.0  1.0  1923 
## prewetness                      0.1  1.0  1808 
## echomotionstationary            0.0  1.0  1931 
## time                            0.0  1.0  2505 
## seedingyes:sne                  0.0  1.0  1799 
## seedingyes:cloudcover           0.0  1.0  1569 
## seedingyes:prewetness           0.1  1.0  1830 
## seedingyes:echomotionstationary 0.1  1.0  2216 
## sigma                           0.0  1.0  1161 
## mean_PPD                        0.0  1.0  3840 
## log-posterior                   0.1  1.0   626 
## 
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample size, and Rhat is th
e potential scale reduction factor on split chains (at convergence Rhat=1).
```

Exercise 2: How do the estimated coefficients compare in this glm model to those from the model fit using lm?

- lm has larger value in positive coefficients and has smaller value in negative coefficients (which means it's more extreme and more overfitting to the data than stan.glm.)

Exercise 3: How do the credible intervals and standard errors of the coeffients compare?

```
# 95% CI comparison
round(confint(ols, level=0.95),3)
```

```
##                                 2.5 % 97.5 %
## (Intercept)                    -6.369  5.676
## seedingyes                      6.077 25.289
## sne                            -1.405  2.244
## cloudcover                     -0.083  0.859
## prewetness                     -3.671 11.888
## echomotionstationary           -1.022  7.328
## time                           -0.099  0.009
## seedingyes:sne                 -5.935 -0.460
## seedingyes:cloudcover          -1.007  0.035
## seedingyes:prewetness         -12.237  7.123
## seedingyes:echomotionstationary  -6.275  5.150
```

```
round(posterior_interval(stan.glm, prob = 0.95), 3)
```

```
##                                  2.5%  97.5%
## (Intercept)                    -4.195  7.312
## seedingyes                      2.355 20.648
## sne                            -1.777  1.698
## cloudcover                     -0.086  0.773
## prewetness                     -3.311 11.154
## echomotionstationary           -1.024  6.381
## time                           -0.092  0.013
## seedingyes:sne                 -4.730  0.393
## seedingyes:cloudcover          -0.880  0.071
## seedingyes:prewetness         -11.715  5.737
## seedingyes:echomotionstationary  -5.076  4.866
## sigma                           1.636  3.497
```

- In general, the standard errors of stan.glm is bigger than lm. However, credible intervals of stan.glm and lm are very similar.

2. Logistic Regression

```
seed <- 196
admissions <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
## view the first few rows of the data
# head(admissions)
admissions$rank <- factor(admissions$rank)
admissions$admit <- factor(admissions$admit)
admissions$gre <- scale(admissions$gre)
p <- 5
n <- nrow(admissions)
```

Frequentist Approach

```
freq.mod <- glm(admit ~. , data = admissions,
                family = binomial())
summary(freq.mod)
```

```
##
## Call:
## glm(formula = admit ~ ., family = binomial(), data = admissions)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -2.6592     1.1657  -2.281 0.022535 *
## gre           0.2616     0.1264   2.070 0.038465 *
## gpa           0.8040     0.3318   2.423 0.015388 *
## rank2        -0.6754     0.3165  -2.134 0.032829 *
## rank3        -1.3402     0.3453  -3.881 0.000104 ***
## rank4        -1.5515     0.4178  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

Exercise 4: Interpret the significant coefficients (at the 0.05 significance level).

- gre: Holding all else constant, for individual who is in rank 1, increasing gre by 1 unit is expected to increase the log-odds of being admitted by 0.2616.
- gpa: Holding all else constant, for individual who is in rank 1, increasing gpa by 1 unit is expected to increase the log-odds of being admitted by 0.8040.
- rank2: Holding all else constant, for individual who switch from rank 1 to rank 2, log-odds of being admitted is decreased by 0.6754.
- rank3: Holding all else constant, for individual who switch from rank 1 to rank 3, log-odds of being admitted is decreased by 1.3402.
- rank4: Holding all else constant, for individual who switch from rank 1 to rank 5, log-odds of being admitted is decreased by 1.5515.

Weakly Informative Prior: Normal

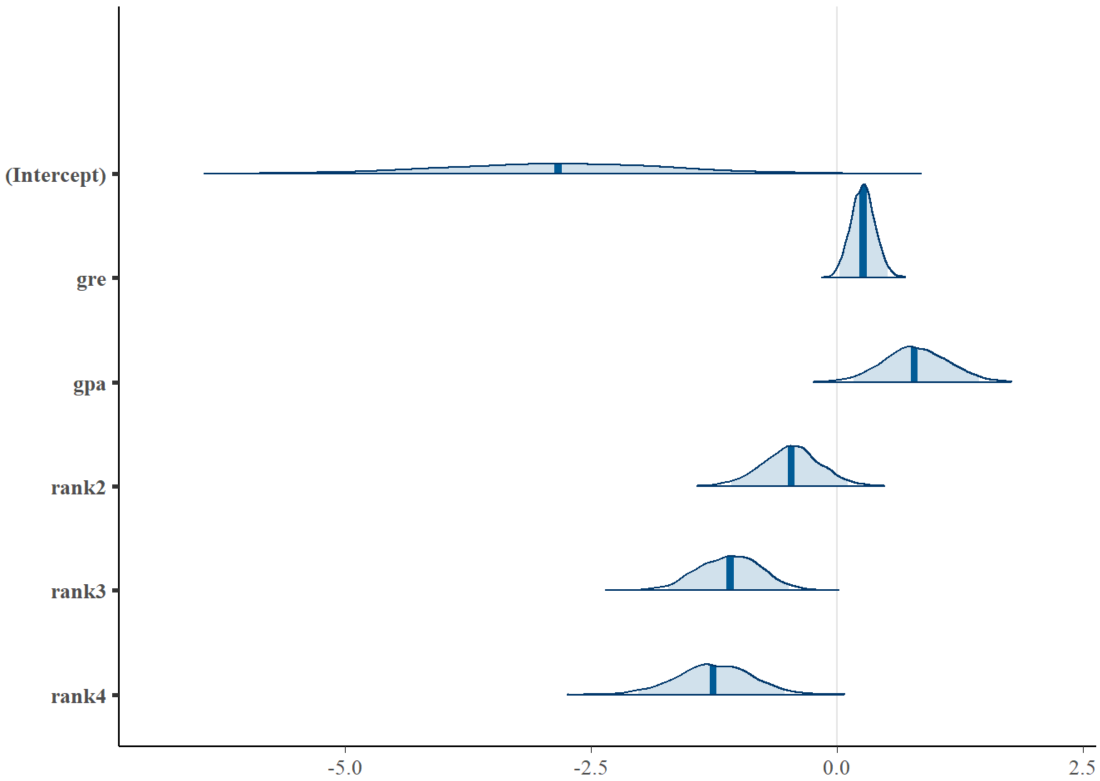
```
post1 <- stan_glm(admit ~ ., data = admissions,
                 family = binomial(link = "logit"),
                 prior = normal(0,1), prior_intercept = normal(0,1),
                 seed = seed,
                 refresh = 0)
summary(post1)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      admit ~ .
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 400
##  predictors:   6
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) -2.8    1.1 -4.3  -2.8  -1.4
## gre          0.3    0.1  0.1   0.3   0.4
## gpa          0.8    0.3  0.4   0.8   1.2
## rank2       -0.5    0.3 -0.8  -0.5  -0.1
## rank3       -1.1    0.3 -1.5  -1.1  -0.7
## rank4       -1.2    0.4 -1.7  -1.3  -0.8
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.3    0.0  0.3   0.3   0.4
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for details see help('summa
ry.stanreg')).
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  3128
## gre           0.0  1.0  2999
## gpa           0.0  1.0  3199
## rank2         0.0  1.0  1752
## rank3         0.0  1.0  1597
## rank4         0.0  1.0  2337
## mean_PPD      0.0  1.0  4222
## log-posterior 0.0  1.0  1844
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample size, and Rhat is th
e potential scale reduction factor on split chains (at convergence Rhat=1).
```

Exercise 5: What do our choice of priors say about our beliefs? How do we interpret these normal priors?

- Choosing $N(0,1)$ means we believe that model coefficients/intercept are as likely to be positive as they are to be negative but they are highly unlikely to be far from zero.

```
mcmc_areas(as.matrix(post1), prob = 0.95, prob_outer = 1)
```




```
round(coef(post1), 3)
```

```
## (Intercept)         gre         gpa       rank2       rank3       rank4
##      -2.832       0.266       0.790      -0.462      -1.084      -1.252
```

```
round(posterior_interval(post1, prob = 0.95), 3)
```

```
##               2.5%  97.5%
## (Intercept) -5.156 -0.602
## gre          0.026  0.515
## gpa          0.173  1.443
## rank2       -1.066  0.116
## rank3       -1.712 -0.487
## rank4       -2.014 -0.531
```

```
round(confint(freq.mod, level=0.95),3) # comparison with the glm model
```

```
## Waiting for profiling to be done...
```

```
##              2.5 % 97.5 %
## (Intercept) -4.978 -0.397
## gre          0.016  0.512
## gpa          0.160  1.464
## rank2       -1.301 -0.057
## rank3       -2.028 -0.670
## rank4       -2.400 -0.754
```

Exercise 6: How do the estimated coefficients compare in this model to those from the model fit using glm?

- The estimated coefficients of stan_glm are very similar (or slightly pulled towards 0) to those of glm since the bayesian regression model is using a weakly informative prior and data outweights the prior belief.

Exercise 7: How do the credible intervals and standard errors of the coefficients compare to the confidence intervals and standard errors from the model fit using glm?

- The credible intervals and standard errors of stan_glm are also very similar to those of glm because of the same reasons above (data outweights the prior belief).

Posterior predictive checks

```
(loo1 <- loo(post1, save_psis = TRUE))
```

```
##
## Computed from 4000 by 400 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -235.3  8.6
## p_loo         5.6  0.3
## looic       470.6 17.2
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
post0 <- stan_glm(admit ~ 1, data = admissions,
                 family = binomial(link = "logit"),
                 prior = normal(0,1), prior_intercept = normal(0,1),
                 seed = seed,
                 refresh = 0)
(loo0 <- loo(post0, save_psis = T))
```

```
##
## Computed from 4000 by 400 log-likelihood matrix
##
##          Estimate  SE
## elpd_loo   -394.4 0.0
## p_loo       117.3 0.0
## looic       788.9 0.0
## ------
## Monte Carlo SE of elpd_loo is 0.5.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

Exercise 8: Which model is better? Why?

- post1 is better than post0 since it has all covariates. This means that covariates contain useful information for predictions.

```
preds <- posterior_linpred(post1, transform=TRUE)
pred <- colMeans(preds)

# classification accuracy
pr <- as.integer(pred >= 0.5)
round(mean(xor(pr,as.integer(admissions$admit==0))),3)
```

```
## [1] 0.705
```

The Horseshoe Prior

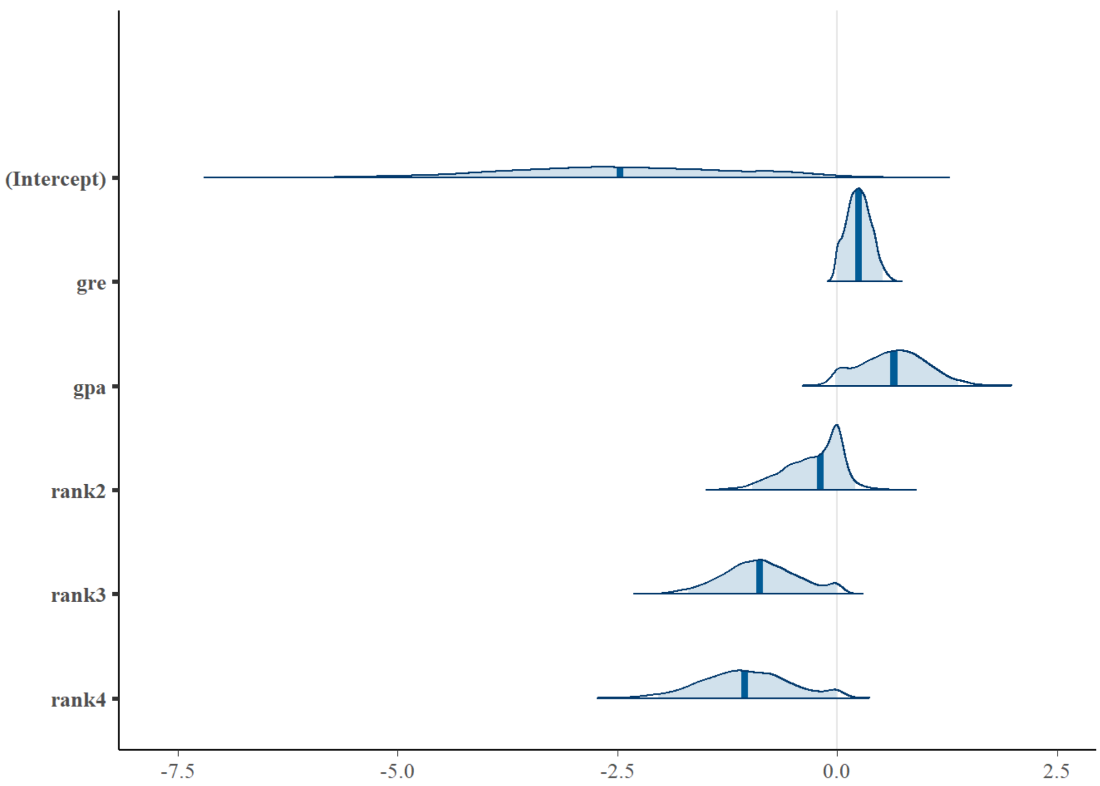
```
p0 <- 2 # prior guess for the number of relevant variables
tau0 <- p0/(p-p0) * 1/sqrt(n) # recommended by Pilronen and Vehtari (2017)
hs_prior <- hs(df=1, global_df=1, global_scale=tau0)
post2 <- stan_glm(admit ~ ., data = admissions,
                 family = binomial(link = "logit"),
                 prior = hs_prior, prior_intercept = normal(0,1),
                 seed = seed,
                 refresh = 0)

round(coef(post2), 3)
```

```
## (Intercept)         gre         gpa       rank2       rank3       rank4
##      -2.472       0.242       0.644      -0.191      -0.883      -1.051
```

```
round(posterior_interval(post2, prob = 0.95), 3)
```

```
##               2.5%  97.5%
## (Intercept) -5.015 -0.189
## gre         -0.003  0.517
## gpa         -0.020  1.375
## rank2       -0.967  0.208
## rank3       -1.697 -0.003
## rank4       -2.035 -0.008
```

```
mcmc_areas(as.matrix(post2), prob = 0.95, prob_outer = 1)
```



Exercise 9: How does posterior inference for the coefficients compare to when we used the weakly informative Normal prior above?

- The posterior inference for the coefficients are more pulled towards 0 when using horseshow prior than coefficients using weakly informative normal prior. Since Horseshoe places higher prior density on 0.

Exercise 10: How do the two models compare in terms of predictive performance? Consider using the loo function as we have been doing.

```
(loo2 <- loo(post2, save_psis = T))
```

```
##
## Computed from 4000 by 400 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo   -238.1  8.2
## p_loo         7.3  0.4
## looic       476.1 16.4
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
```

```
rstanarm::compare_models(loo1, loo2)
```

```
## Warning: 'rstanarm::compare_models' is deprecated.
## Use 'loo_compare' instead.
## See help("Deprecated")
```

```
## Warning: 'loo::compare' is deprecated.
## Use 'loo_compare' instead.
## See help("Deprecated")
```

```
## Model formulas:
##  :  NULL
##  :  NULLelpd_diff        se
##      -2.8       1.0
```

- The model using weakly informative normal prior performs better than model using horseshoe prior since elpd_diff is negative (favors first model). Also since this dataset has n>>p (number of observations greater than number of parameters), meaning that it doesn't quite make sense to use horseshoe prior which tends to shrink all coefficients towards 0.