HW 1

Reese Mullen

9-12-24

`{r setup, include=FALSE} knitr::opts_chunk$set(echo = TRUE) pacman::p_load("bayesplot","knitr","arm","g`

7.2 Fake-data simulation and regression:

Simulate 100 data points from the linear model, $y = a + bx +$ error, with $a = 5$, $b = 7$, the values of x being sampled at random from a uniform distribution on the range $[0, 50]$, and errors that are normally distributed with mean 0 and standard deviation 3.

7.2a

Fit a regression line to these data and display the output.

`{r} x1<-runif(100,0,50) e1<-rnorm(100,0,3) y1<-5+7*x1+e1 lm_1<-lm(y1~x1) summary(lm_1)`

7.2b

Graph a scatterplot of the data and the regression line.

`{r} plot(x1,y1) abline(lm_1,col ="red")`

7.2c

Use the `text` function in R to add the formula of the fitted line to the graph.

`{r} plot(x1,y1) abline(lm_1,col ="red") text(x=15, y =170, "y = 5 + 7x + error")`

7.3 Fake-data simulation and fitting the wrong model:

Simulate 100 data points from the model $y = a + bx + cx^2 +$ error, with the values of x being sampled at random from a uniform distribution on the range $[0, 50]$, errors that are normally distributed with mean 0 and standard deviation 3, and a, b, c chosen so that a scatterplot of the data shows a clear nonlinear curve.

7.3 a

Fit a regression line `stan_glm(y ~ x)` to these data and display the output.

`{r} x2 <-runif(100, 0, 50) e2 <-rnorm(100, 0, 3) y2<-10+1*x2+-0.1*x2^2+e2 stan_glm1<-rstanarm::stan_glm`
`summary(stan_glm1)`

7.3b

Graph a scatterplot of the data and the regression line. This is the best-fit linear regression. What does "best-fit" mean in this context? Best fit in this context refers to closest linear approximation of the model although that does not define the true shape of our data.

`{r} plot(x2,y2) abline(stan_glm1, col = "red")`

7.6 Formulating comparisons as regression models:

Take the election forecasting model and simplify it by creating a binary predictor defined as $x = 0$ if income growth is less than 2% and $x = 1$ if income growth is more than 2%.

"'{r} elect <- read.table("~/Downloads/hibbs.dat", header = TRUE) x7.6 <- ifelse(elect$growth > 2, 1, 0)

~~~
### 7.6a
Compute the difference in incumbent party's vote share on average, comparing those two  groups of elect

```{r}
incumb_over<- subset(elect, x7.6 ==1)
incumb_under<- subset(elect, x7.6 == 0)
mean_over<-mean(incumb_over$vote)
mean_under<-mean(incumb_under$vote)

diff<- mean_over-mean_under

se_over<-sd(incumb_over$vote)/sqrt(nrow(incumb_over))
se_under<-sd(incumb_under$vote)/sqrt(nrow(incumb_under))
~~~

7.6b

Regress incumbent party's vote share on the binary predictor of income growth and check that the resulting estimate and standard error are the same as above.

Regression has a coefficient of 5.51 and difference of means was 5.51. `{r} model7.6<-lm(elect$vote~x7.6) summary(model7.6)`

8.8 Comparing lm and stan_glm:

Use simulated data to compare least squares estimation to default Bayesian regression:

8.8a

Simulate 100 data points from the model, $y = 2 + 3x$ + error, with predictors x drawn from a uniform distribution from 0 to 20 and with independent errors drawn from the normal distribution with mean 0 and standard deviation 5. Fit the regression of y on x data using `lm` and `stan_glm` (using its default settings) and check that the two programs give nearly identical results.

`{r} x3<-runif(100,0,20) e3<-rnorm(100,0,5) y3<- 2+3*x3+e3 lm_2<-lm(y3~x3) stan_glm2<-rstanarm::stan_glm summary(lm_2) summary(stan_glm2)`

8.8b

Plot the simulated data and the two fitted regression lines.

`{r} plot(x3, y3) abline(lm_2, col = "red", lwd = 2, lty = 1) abline(a = coef(stan_glm2)[1], b = coef(stan_glm2)[2], col = "blue", lwd = 2, lty = 2)`

8.8c

Repeat the two steps above, but try to create conditions for your simulation so that `lm` and `stan_glm` give much different results.

"'{r} x4<-runif(10,0,20) e4<-rnorm(10,0,1000) y4<- 2+3*x4+e4 lm_3<-lm(y4~x4) stan_glm3<-rstanarm::stan_glm(y4~x4) summary(lm_3) summary(stan_glm3)

```
## 10.1 Regression with interactions:
Simulate 100 data points from the model, $y = b_0 + b_1 x +  b_2 z + b_3 x z$ + error, with a continuous

### 10.1a
Display your simulated data as a graph of $y$ vs $x$, using dots and circles for the points with $z$ = 0

'''{r}
b0 <- 1
b1 <- 2
b2 <- -1
b3 <- -2
z<-rbinom(100, 1, 0.5)
x5<-rnorm(100, mean = z, sd = 1)
e5<-rnorm(100, mean = 0, sd = 3)
y5 <- b0 + b1 * x5 + b2 * z + b3 * x5 * z + e5

Q10.1<-data.frame(x = x5, y = y5, z = as.factor(z))
ggplot(Q10.1, aes(x = x, y = y, shape = z)) +
  geom_point(size = 2) +
  scale_shape_manual(values = c(16, 1)) +
  theme_minimal()
```

10.1b

Fit a regression predicting y from x and z with no interaction. Make a graph with the data and two parallel lines showing the fitted model.

"'{r} model10.1b <- lm(y5 ~ x5 + z, data = Q10.1) summary(model10.1b)

Q10.1$pred1<- predict(model10.1b) ggplot(Q10.1, aes(x = x5, y = y5, shape = z, color = z)) + geom_point(size = 2) + geom_smooth(method = "lm", aes(y = pred1), se = FALSE) + scale_shape_manual(values = c(16, 1)) + theme_minimal()

```
### 10.1c
Fit a regression predicting $y$ from $x$, $z$, and their interaction. Make a graph with the data and tw

'''{r}
```

```
model10.1c <- lm(y5 ~ x5 * z, data = Q10.1)
summary(model10.1c)

Q10.1$pred2<- predict(model10.1c)
ggplot(Q10.1, aes(x = x5, y = y5, shape = z, color = z)) +
  geom_point(size = 2) +
  geom_smooth(method = "lm", aes(y = pred2), se = FALSE) +
  scale_shape_manual(values = c(16, 1)) +
  theme_minimal()
```

10.2 Regression with interactions:

Here is the output from a fitted linear regression of outcome y on pre-treatment predictor x, treatment indicator z, and their interaction:

““{verbatim} Mediam MAD_SD (Intercept) 1.2 0.2 x 1.6 0.4 z 2.7 0.3 x:z 0.7 0.5

Auxiliary parameter(s): Median MAD_SD sigma 0.4 0.0 ““

10.2a

Write the equation of the estimated regression line of y on x for the treatment group and the control group, and the equation of the estimated regression line of y on x for the control group.

“‘{r} #Control Group #$y = 1.2 + 1.6\ x$ + error

#Treatment Group # $y = 3.9 + 2.3\ x$ + error

~~~
### 10.2b
Graph with pen on paper the two regression lines, assuming the values of $x$ fall in the range $(0, 10)

```{r}
b0_con<-1.2
b1_con<-1.6
b0_treat<-3.9
b1_treat<-2.3
sigma<-0.4

x10.2<- runif(100, 0, 10)
z10.2 <- rbinom(100, 1, 0.5)
y10.2 <- ifelse(z10.2 == 0,
            b0_con + b1_con * x10.2 + rnorm(100, 0, sigma),
            b0_treat + b1_treat * x10.2 + rnorm(100, 0, sigma))

Q10.2 <- data.frame(x = x10.2, y = y10.2, z = as.factor(z10.2))

ggplot(Q10.2, aes(x = x10.2, y = y10.2, color = as.factor(z10.2), shape = as.factor(z10.2))) +
  geom_point(size = 2) +
  geom_abline(intercept = b0_con, slope = b1_con) +
  geom_abline(intercept = b0_treat, slope = b1_treat,color = "red") +
  scale_shape_manual(values = c(16, 1)) +
  theme_minimal()
~~~

10.5 Regression modeling and prediction:

The folder `KidIQ` contains a subset of the children and mother data discussed earlier in the chapter. You have access to children's test scores at age 3, mother's education, and the mother's age at the time she gave birth for a sample of 400 children.

`{r} KidIQ<-read.csv("child_iq.csv")`

10.5a

Fit a regression of child test scores on mother's age, display the data and fitted model, check assumptions, and interpret the slope coefficient. Based on this analysis, when do you recommend mothers should give birth? What are you assuming in making this recommendation?

The initial model has a Y intercept of 67.78 for the childrens' test scores. However, this does not make sense because the mother's in our data are all between the age of 17 and 29. When we center the data around the mean the intercept becomes 86.93. The slope stays the same at 0.84 increase for each year older that the mother is when she gives birth. The recommendation would be for the mother to give birth at age 29 as this is the highest datapoint that our model predicts. This model assumes that other factors such as education have no impact on the childrens' testing. "'{r} model10.5a1<-lm(ppvt~momage, data = KidIQ) summary(model10.5a1)

mean(KidIQ$momage)KidIQ$cent_momage<-KidIQ$momage - mean(KidIQ$momage)

model10.5a2<-lm(ppvt~cent_momage, data = KidIQ) summary(model10.5a2)

```
### 10.5b
Repeat this for a regression that further includes mother's education, interpreting both slope coeffici
For each year older that the mother is when she gives borth the child's test score increases by 0.34 poi

'''{r}
model10.5b1<-lm(ppvt~momage+educ_cat, data = KidIQ)
summary(model10.5b1)

model10.5b2<-lm(ppvt~cent_momage+educ_cat, data = KidIQ)
summary(model10.5b2)
```

10.5c

Now create an indicator variable reflecting whether the mother has completed high school or not. Consider interactions between high school completion and mother's age. Also create a plot that shows the separate regression lines for each high school completion status group.

"'{r} KidIQ$MomHS < -ifelse(KidIQ$educ_cat >=2, 1, 0) model10.5c <- lm(ppvt ~ momage * MomHS, data = KidIQ)

summary(model10.5c) ggplot(KidIQ, aes(x = momage, y = ppvt, color = as.factor(MomHS))) + geom_point() + geom_smooth(method = "lm", se = FALSE) + theme_minimal()

```
### 10.5d
Finally, fit a regression of child test scores on mother's age and education level for the first 200 chi

'''{r}
```

```
train<-KidIQ[1:200,]
test<-KidIQ[201:400,]

train_mod<-lm(ppvt~momage+MomHS, data = train)
pred3<-predict(train_mod, newdata = test)
Q10.5 <-data.frame(real = test$ppvt, pred = pred3)

ggplot(Q10.5, aes(x = real, y = pred3)) +
  geom_point() +
  geom_abline(slope =1, linetype = "dashed")+
  theme_minimal()
```

10.6 Regression models with interactions:

The folder `Beauty` contains data (use file `beauty.csv`) from Hamermesh and Parker (2005) on student evaluations of instructors' beauty and teaching quality for several courses at the University of Texas. The teaching evaluations were conducted at the end of the semester, and the beauty judgments were made later, by six students who had not attended the classes and were not aware of the course evaluations.

See also Felton, Mitchell, and Stinson (2003) for more on this topic.

`{r} beauty<-read.csv("~/Downloads/beauty.csv")`

10.6a

Run a regression using beauty (the variable `beauty`) to predict course evaluations (`eval`), adjusting for various other predictors. Graph the data and fitted model, and explain the meaning of each of the coefficients along with the residual standard deviation. Plot the residuals versus fitted values. Increasing the beauty rating by 1unit increases the evaluation by 0.14 on average. If the instructor is a minority the evaluation decreases by 0.06 on average. For every year older the instructor is, the evaluation decreases by 0.002. If the instructor is not female the evaluation decreases by 0.197 on average. If the instructor is a nonenglish speaker, the evaluation decreases by 0.27 on average. If the course was lower the evaluation increased by 0.0.98 on average. For each increase in the course id the evaluation decreased by 0.0003 on average. The residuals are fairly normally distributed although there is a less density at the lower end. "'{r} model10.6a <- rstanarm::stan_glm(eval ~ beauty + minority + age + female+ nonenglish+ lower + course_id, data = beauty) summary(model10.6a)

ggplot(beauty, aes(x = beauty, y = eval)) + geom_point() + geom_smooth(method = "lm", se = FALSE) + theme_minimal()

plot(model10.6a, which = 1)

res_sd<-summary(model10.6a$sigma) res_sd

```
### 10.6b
Fit some other models, including beauty and also other predictors. Consider at least one model with int

For the first model, each increase in beauty rating increases the evaluation by 0.19 on average. If the

For the second model, an increase on the beauty scale increases the evaluation by 0.17 on average. If t

For the third model, each increase in beauty rating increases the evaluation by 0.2 on average. If the
'''{r}
model10.6b1 <- lm(eval ~ beauty*female + age, data = beauty)
```

```
summary(model10.6b1)
model10.6b2 <- lm(eval ~ beauty*minority + age, data = beauty)
summary(model10.6b2)
model10.6b3<-rstanarm::stan_glm(eval ~ beauty*female + age+nonenglish, data = beauty)
summary(model10.6b3)
```

10.7 Predictive simulation for linear regression:

Take one of the models from the previous exercise.

10.7a

Instructor A is a 50-year-old woman who is a native English speaker and has a beauty score of -1. Instructor B is a 60-year-old man who is a native English speaker and has a beauty score of -0.5. Simulate 1000 random draws of the course evaluation rating of these two instructors. In your simulation, use `posterior_predict` to account for the uncertainty in the regression parameters as well as predictive uncertainty.

```
{r} Q10.7a<-data.frame(beauty = c(-1, -0.5),   age = c(50, 60),   female = c(1, 0),
nonenglish = c(0,0)) pred4<-rstanarm::posterior_predict(model10.6b3, newdata = Q10.7a,
nsamples = 1000)
```

10.7b

Make a histogram of the difference between the course evaluations for A and B. What is the probability that A will have a higher evaluation?

"'{r} pred_df<-as.data.frame(pred4) names(pred_df)<-c("inst_A", "inst_B")

pred_df$diff < -pred_d f$inst_A - pred_df$inst_B

ggplot(pred_df, aes(x = diff)) + geom_histogram(bins = 30) + theme_minimal()

prob_A<- mean(pred_df$diff > 0) prob_A

```
## 10.8 How many simulation draws:
Take the model from Exercise 10.6 that predicts course evaluations from beauty and other predictors.

### 10.8a
Display and discuss the fitted model. Focus on the estimate and standard error for the coefficient of b
The coefficient is 0.1 for beauty, which is rather small, but the standard error is much smaller, which
'''{r}
summary(model10.6a)
```

10.8b

Compute the median and mad sd of the posterior simulations of the coefficient of beauty, and check that these are the same as the output from printing the fit.

"'{r} post_sample<-as.matrix(model10.6a) beauty_sample<-post_sample[, "beauty"] median_beauty <- median(beauty_sample) mad_beauty <- mad(beauty_sample)

print(median_beauty) print(mad_beauty)

```
### 10.8c
Fit again, this time setting ‘iter‘ = 1000 in your ‘stan_glm‘ call. Do this a few times in order to get

‘‘‘{r}
fit_model <- function(data) {
  model <- stan_glm(
    eval ~ beauty + minority + age + female + nonenglish + lower + course_id,
    data = data,
    iter = 1000
  )
  post_sample <- as.matrix(model)
  return(post_sample[, "beauty"])
}
n_repeats<-3
all_samples<-lapply(1:n_repeats, function(x) fit_model(beauty))

samples_matrix<- do.call(cbind, all_samples)

median_samples <- apply(samples_matrix, 2, median)
mad_samples <- apply(samples_matrix, 2, mad)
mean_sd_samples <- apply(samples_matrix, 2, sd)
print(c(median_samples, mad_samples, mean_sd_samples))
```

10.8d

Repeat the previous step, setting `iter` = 100 and then `iter` = 10.

“‘{r} fit_model <- function(data) { model <- stan_glm(eval ~ beauty + minority + age + female + nonenglish + lower + course_id, data = data, iter = 100) post_sample <- as.matrix(model) return(post_sample[, “beauty”]) } n_repeats<-3 all_samples<-lapply(1:n_repeats, function(x) fit_model(beauty))

samples_matrix<- do.call(cbind, all_samples)

median_samples <- apply(samples_matrix, 2, median) mad_samples <- apply(samples_matrix, 2, mad) mean_sd_samples <- apply(samples_matrix, 2, sd) print(c(median_samples, mad_samples, mean_sd_samples))

fit_model <- function(data) { model <- stan_glm(eval ~ beauty + minority + age + female + nonenglish + lower + course_id, data = data, iter = 10) post_sample <- as.matrix(model) return(post_sample[, “beauty”]) } n_repeats<-3 all_samples<-lapply(1:n_repeats, function(x) fit_model(beauty))

samples_matrix<- do.call(cbind, all_samples)

median_samples <- apply(samples_matrix, 2, median) mad_samples <- apply(samples_matrix, 2, mad) mean_sd_samples <- apply(samples_matrix, 2, sd) print(c(median_samples, mad_samples, mean_sd_samples)) “‘

10.8e

How many simulations were needed to give a good approximation to the mean and standard error for the coefficient of beauty? 1000