HW5

Rory Michelen

October 30, 2019

Question 1

The data for both training and testing is provided below:

```
set.seed(1)
x1 <- runif(20)
x2 <- floor(10 * runif(20)) + 1
Get.y<-function(x1,x2){
  return(2 + 6 * x1 - 0.5 * x2 + 0.8*rnorm(length(x1)))
}

y <- 2 + 6 * x1 - 0.5 * x2 + 0.8*rnorm(length(x1))

x1.test <- runif(20)
x2.test <- floor(10 * runif(20)) + 1
```

Next, a bayesian linear regression model is built by calling openBugs from R using R2 Open Bugs

```
model <- function() {
  for(i in 1:20){
    y[i]~dnorm(mu[i],tau)
    mu[i]<-beta0+beta1*x1[i]+beta2*x2[i]
  }
  beta0~dnorm(0,0.001)
  beta1~dnorm(0,0.001)
  beta2~dnorm(0,0.001)
  tau~dgamma(0.01,0.01)
}

model.file <- file.path(tempdir(), "model.txt")
write.model(model, model.file)

data<-list("x1","x2","y")
inits <- function() { list(beta0=0,beta1=0,beta2=0,tau=1) }
params<-c("beta0","beta1","beta2","tau")

out <- bugs(data, inits, params,model.file, n.iter=100000)

summary<-data.frame(out$summary)%>%
  select(mean)%>%
  rownames_to_column("Param")%>%
  spread(key="Param",value="mean")
```

For this particular set of randomly generated data, we get a mean squared error of 7.04. Additionally, a plot is provided that demonstrates how each parameter varies from the true value of the parameter.

Also provided is a comparison of a frequentist linear model trained and tested on the same data. Since we used non-informative priors, our estimates for each paramter are approximately equal

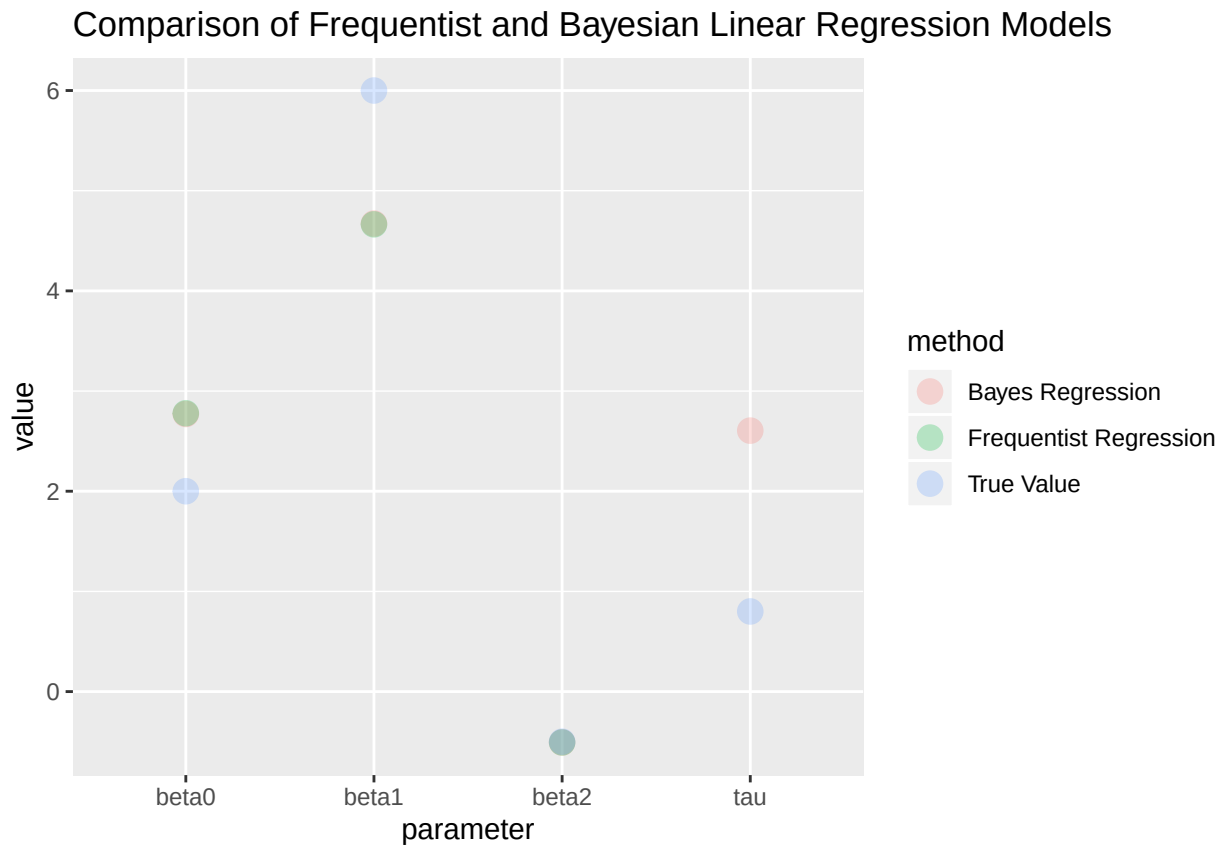
```
data.frame(y)%>%
  mutate(y_bar=with(beta0+beta1*x1.test+beta2*x2.test,data=summary))%>%
  mutate(squared.error=(y-y_bar)^2)%>%
  summarise(mse=mean(squared.error))%>%
  kable()
```

```
## Warning: package 'bindrcpp' was built under R version 3.5.1
```

mse
7.038685

```
compare<-rbind(summary,c(lm(y~x1+x2)$coefficients,NA,NA),c(2,6,-.5,NA,0.8))%>%
  add_column(method=c("Bayes Regression","Frequentist Regression","True Value"))

compare%>%
  select(method,beta0,beta1,beta2,tau)%>%
  gather(key="parameter",value="value",beta0,beta1,beta2,tau)%>%
  ggplot(aes(x=parameter,y=value,col=method))+
    geom_point(size=4,alpha=0.25)+
    labs(title='Comparison of Frequentist and Bayesian Linear Regression Models')
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



Question 2

Part A

While a starting odc file was provided, I am a bit strongheaded and wanted to practice usng R2 open bugs. For that reason, I took the original odc file and create a (1) csv containing the empirical data and (2) an R2open bugs model provided below.

For the second part of this question, I enumerated overall all potential models and selected the one with the best R-Squared value. The formulation of the models is the following

Model j: $bf = b_{0,j} + b_{1,j} * age + b_{2,j} * bai + b_{3,j} * bmi + b_{4,j} * bb + b5, j * gender + e_j$

For models j=2,3,4,5,6 and $b_{i,j} = 0$ if both of the conditions are true: $i > 0$ and $i \neq j$. For model 1, all parameters can be nonzero.

```
# Data input
q2.data<-read_csv("q2_input.csv")
Age<-q2.data$Age
Gender<-q2.data$Gender
BAI<-q2.data$BAI
BMI<-q2.data$BMI
BF<-q2.data$BF
N<-nrow(q2.data)
```

Below is the cretion of all 6 models to be considered.

```
#Open Bugs Model creation
# Model 1: All parameters included
q2.model.1<-function(){
for(i in 1:N){
  BF[i] ~ dnorm(mu[i], tau)
  BB[i] <- BAI[i] * BMI[i]
  mu[i] <- b0 + b1 * Age[i] + b2*BAI[i] + b3*BMI[i] + b4*BB[i] + b5* Gender[i]
}

b0 ~ dnorm(0, 0.001)
b1 ~ dnorm(0, 0.001)
b2 ~ dnorm(0, 0.001)
b3 ~ dnorm(0, 0.001)
b4 ~ dnorm(0, 0.001)
b5 ~ dnorm(0, 0.001)
tau ~ dgamma(0.001, 0.001)
}

# Model 2: Age is the only independent variable
q2.model.2<-function(){
for(i in 1:N){
BF[i] ~ dnorm(mu[i], tau)
BB[i] <- BAI[i] * BMI[i]
mu[i] <- b0 + b1 * Age[i]
}

b0 ~ dnorm(0, 0.001)
b1 ~ dnorm(0, 0.001)
tau ~ dgamma(0.001, 0.001)
}
```

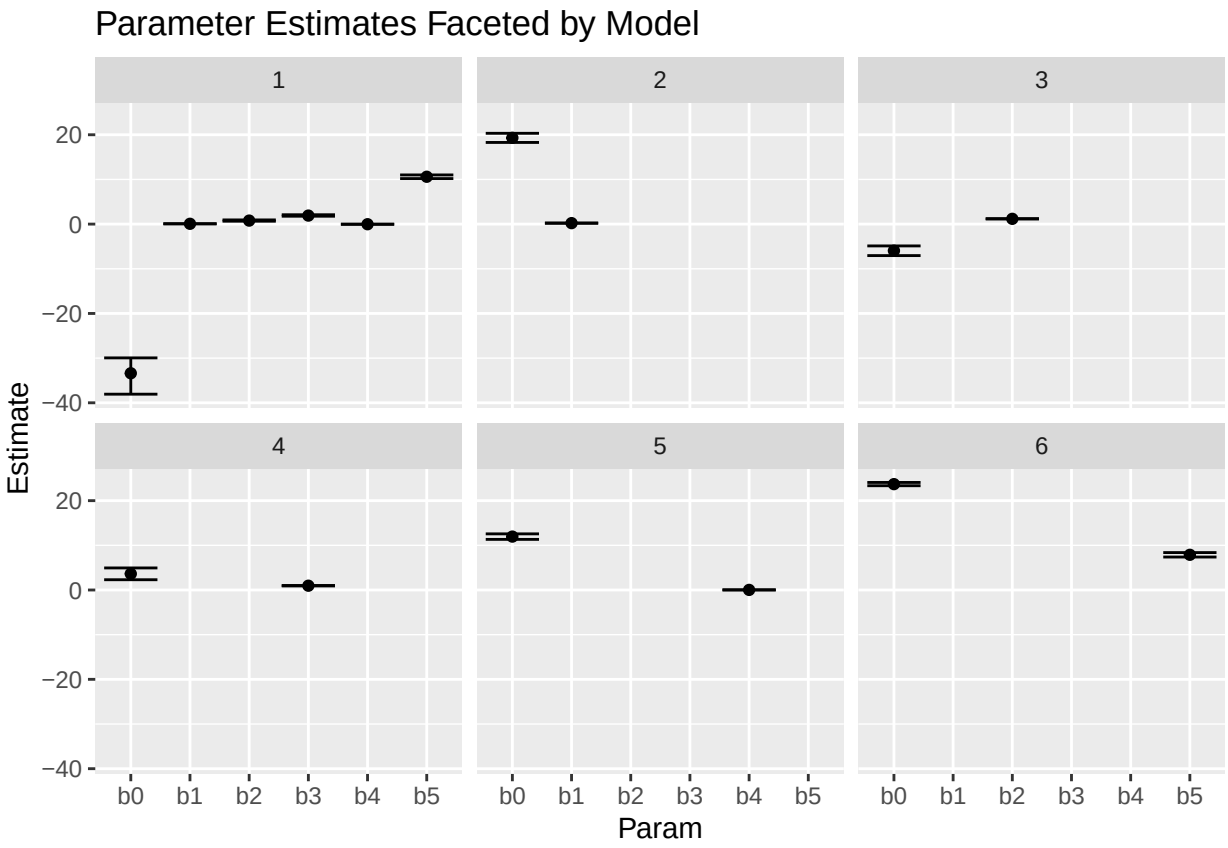
```
# Model 3: BAI is the only independent variable
q2.model.3<-function(){
for(i in 1:N){
BF[i] ~ dnorm(mu[i], tau)
BB[i] <- BAI[i] * BMI[i]
mu[i] <- b0 + b2 * BAI[i]
}

b0 ~ dnorm(0, 0.001)
b2 ~ dnorm(0, 0.001)
tau ~ dgamma(0.001, 0.001)
}

# Model 4: BMI is the only independent variable
q2.model.4<-function(){
for(i in 1:N){
BF[i] ~ dnorm(mu[i], tau)
BB[i] <- BAI[i] * BMI[i]
mu[i] <- b0 + b3 * BMI[i]
}

b0 ~ dnorm(0, 0.001)
b3 ~ dnorm(0, 0.001)
tau ~ dgamma(0.001, 0.001)
}

# Model 5: BB is the only independent variable
q2.model.5<-function(){
for(i in 1:N){
BF[i] ~ dnorm(mu[i], tau)
BB[i] <- BAI[i] * BMI[i]
mu[i] <- b0 + b4 * BB[i]
}

b0 ~ dnorm(0, 0.001)
b4 ~ dnorm(0, 0.001)
tau ~ dgamma(0.001, 0.001)
}

# Model 6: Gender is the only independent variable
q2.model.6<-function(){
for(i in 1:N){
BF[i] ~ dnorm(mu[i], tau)
BB[i] <- BAI[i] * BMI[i]
mu[i] <- b0 + b5 * Gender[i]
}

b0 ~ dnorm(0, 0.001)
b5 ~ dnorm(0, 0.001)
tau ~ dgamma(0.001, 0.001)
}

params.1<-c("b0","b1","b2","b3","b4","b5")
```

```
inits<-function(){list(b0=1,b1=0,b2=0,b3=0,b4=0,b5=0,tau=1)}

data<-list("Age","BAI","BMI","Gender","BF","N")

model.file <- file.path(tempdir(), "model.txt")

parse.output<-function(out,model.number){
  output<-out$summary%>%
    data.frame()%>%
    rownames_to_column("Param")%>%
    dplyr::filter(Param!='deviance')%>%
    mutate(model.number=model.number)

  return(output)
}

write.model(q2.model.1, model.file)
params.1<-c("b0","b1","b2","b3","b4","b5")
out.1 <- bugs(data, inits, params.1,model.file, n.iter=10000)

write.model(q2.model.2, model.file)
params.2<-c("b0","b1")
out.2 <- bugs(data, inits, params.2,model.file, n.iter=10000)

write.model(q2.model.3, model.file)
params.3<-c("b0","b2")
out.3 <- bugs(data, inits, params.3,model.file, n.iter=10000)

write.model(q2.model.4, model.file)
params.4<-c("b0","b3")
out.4 <- bugs(data, inits, params.4,model.file, n.iter=10000)

write.model(q2.model.5, model.file)
params.5<-c("b0","b4")
out.5 <- bugs(data, inits, params.5,model.file, n.iter=10000)

write.model(q2.model.6, model.file)
params.6<-c("b0","b5")
out.6 <- bugs(data, inits, params.6,model.file, n.iter=10000)

summary<-rbind(parse.output(out.1,1),
               parse.output(out.2,2),
               parse.output(out.3,3),
               parse.output(out.4,4),
               parse.output(out.5,5),
               parse.output(out.6,6))
```

Before looking at the R-Squared Values, let's take a look at the parameter estimates for each model. While gender appears to have the most influence over the first model (with all features included), it does not guarentee that it is the best predictor. In fact since it is a binary feature, I expect model 6 (only to containing gender) to perform poorly.

```
# Analysis of results
summary%>%
```

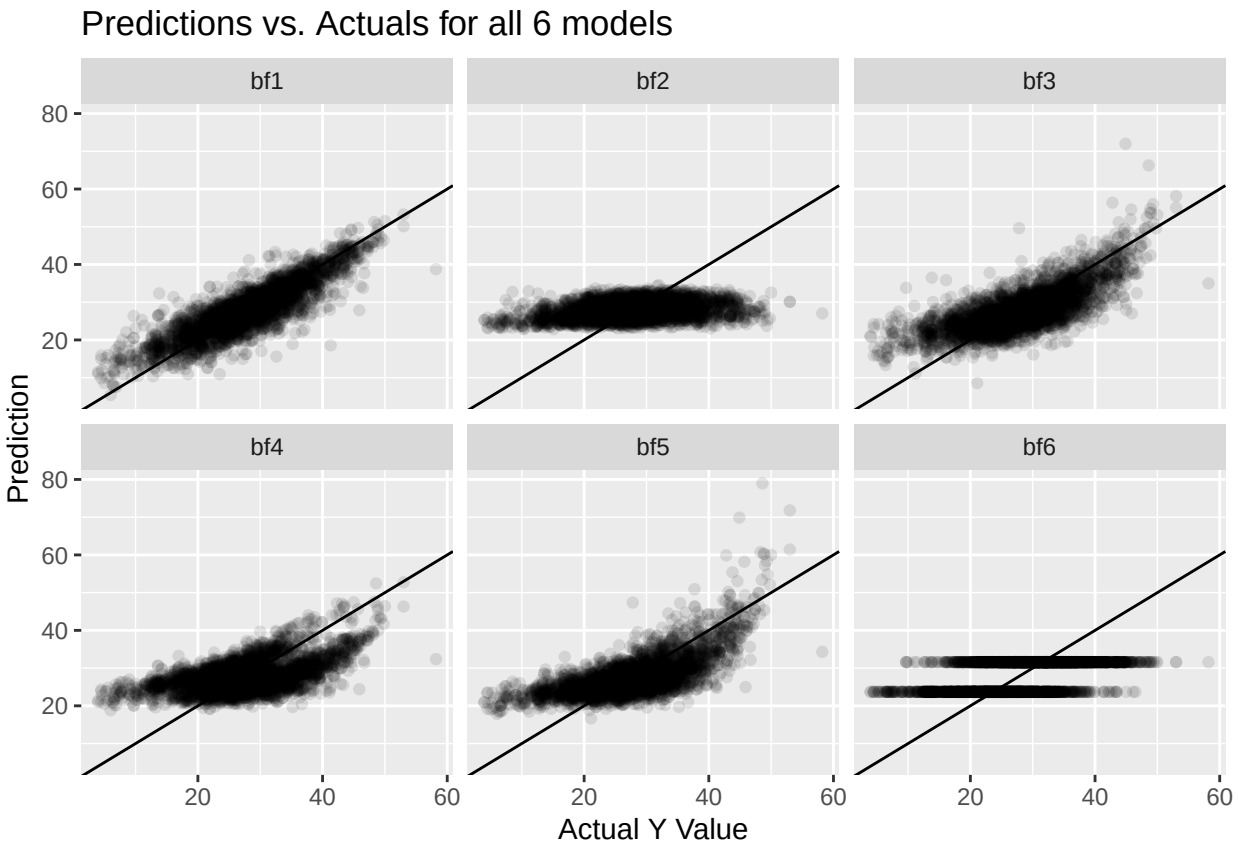
```
  ggplot(aes(x=Param,y=mean))+
    geom_point()+
    facet_wrap(~model.number)+
    geom_errorbar(aes(ymin=X2.5.,ymax=X97.5.))+
    labs(title="Parameter Estimates Faceted by Model",y="Estimate")
```



Next, here is a plot of the actual values of y versus the predictions. As expected, model 6 performs poorly whereas models 1 and 3 appear to perform quite well. However, models 2-6 all appear have non-normal residuals, which is an indicator of poor model performance.

```
summary.wide<-summary%>%
  select(Param,model.number,mean)%>%
  spread(key=Param,value=mean)%>%
  mutate(b0=ifelse(is.na(b0),0,b0),
         b1=ifelse(is.na(b1),0,b1),
         b2=ifelse(is.na(b2),0,b2),
         b3=ifelse(is.na(b3),0,b3),
         b4=ifelse(is.na(b4),0,b4),
         b5=ifelse(is.na(b5),0,b5))

Get.y.pred<-function(modelNum,Age,Gender,BAI,BMI){
  summary.wide.m<-summary.wide%>%
    dplyr::filter(model.number==modelNum)
  return(with(b0+b1*Age+b2*BAI+b3*BMI+b4*BAI*BMI+b5*Gender,data=summary.wide.m))
}
```

```
predictions.added<-q2.data%>%
  mutate(bf1=Get.y.pred(1,Age,Gender,BAI,BMI),
         bf2=Get.y.pred(2,Age,Gender,BAI,BMI),
         bf3=Get.y.pred(3,Age,Gender,BAI,BMI),
         bf4=Get.y.pred(4,Age,Gender,BAI,BMI),
         bf5=Get.y.pred(5,Age,Gender,BAI,BMI),
         bf6=Get.y.pred(6,Age,Gender,BAI,BMI))%>%
  gather(key=model,value=y.pred,bf1,bf2,bf3,bf4,bf5,bf6)

predictions.added%>%
  ggplot(aes(x=BF,y=y.pred))+geom_point(alpha=0.1)+facet_wrap(~model)+geom_abline(slope=1,intercept = 0)
  labs(title='Predictions vs. Actuals for all 6 models',x='Actual Y Value',y='Prediction')
```




```
predictions.added%>%
  group_by(model)%>%
  mutate(y.bar=mean(BF))%>%
  summarise(sse=sum((BF-y.pred)^2),
            ssr=sum((y.pred-y.bar)^2))%>%
  mutate(r.squared=ssr/(ssr+sse))%>%
  select(model,r.squared)%>%
  kable()
```

model	r.squared
bf1	0.7525161
bf2	0.0844493
bf3	0.5470544

model	r.squared
bf4	0.2960905
bf5	0.4759727
bf6	0.2315755

Part A

As expected from our visual inspection, models 1 and 3 have the highest R squared value and will therefore be selected as our final models.

The parameters of these models are below:

```
summary.wide%>%
  dplyr::filter(model.number %in%c(1,3))%>%
  kable()
```

model.number	b0	b1	b2	b3	b4	b5
1	-33.388529	0.0731677	0.7871171	1.908926	-0.0244614	10.60096
3	-5.953853	0.0000000	1.1828034	0.000000	0.0000000	0.00000

Part B

Using the two models selected above, predictions are provided, below:

```
summary.wide%>%
  dplyr::filter(model.number %in%c(1,3))%>%
  mutate(Age=35,BAI=26,BMI=20,Gender=0,BB=520)%>%
  mutate(prediction=b0+b1*Age+b2*BAI+b3*BMI+b4*BB+b5*Gender)%>%
  select(model.number,prediction)%>%
  kable()
```

model.number	prediction
1	15.09599
3	24.79904

Question 3

In this example, we are provided the data provided is of the form: number of success and number of trials. This lends itself to a binomial distribution. The following R2OpenBugs model creates a logit regression

```
# Data entry
responses<-c(0,9,21,47,60,63)
N<-70
current<-c(0,1,2,3,4,5)
N.iter<-6

# Winbugs model specificiation
model<-function(){
  for(i in 1:N.iter){
    responses[i]~dbin(phat[i],N)
    logit(phat[i])<-beta[1]+beta[2]*current[i]
  }
```

```
  beta[1]~dnorm(0,0.001)
  beta[2]~dnorm(0,0.001)
}

model.file <- file.path(tempdir(), "model.txt")
write.model(model, model.file)

data<-list("responses","N","current","N.iter")
inits <- function() {list(beta=c(0,0))}
params<-c("beta")

out <- bugs(data, inits, params,model.file, n.iter=100000)
```

To get the upper and lower bounds of the 95% credible set, I will use the upper and lower bounds of the credible sets on each parameter, respectively. As seen in the code below, the credible set is [19.7%,73.4%]

```
log.odds.to.p<-function(l){
  return(exp(l)/(1+exp(l)))
}

out$summary%>%
  data.frame()%>%
  select(mean,X2.5.,X97.5.)%>%
  rownames_to_column("Param")%>%
  gather(key="param.type",value='value',mean,X2.5.,X97.5.)%>%
  spread(key="Param",value="value")%>%
  mutate(log.odds.estimate=`beta[1]`+`beta[2]`*2.5)%>%
  mutate(p.hat=log.odds.to.p(log.odds.estimate))%>%
  select(param.type,p.hat)%>%
  spread(key="param.type",value="p.hat")%>%
  kable()
```

mean	X2.5.	X97.5.
0.4529487	0.197103	0.7341927