7.1(a)

I used the sample function to generate the follow x values:

```
> sample(1:8,4)
[1] 5 8 4 7
```

I will be using SocSec, SocEnr, Medicare, SupSec as my x-variables. I created a new variable glm.fit2 and created a summary.

```
> glm.fit2 = glm(Vote~SocSec+SocEnr+Medicare+SupSec,data=Hospital,family=binomial)
> summary(glm.fit2)

Call:
glm(formula = Vote ~ SocSec + SocEnr + Medicare + SupSec, family = binomial,
    data = Hospital)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9088  -0.8811   0.3720   0.9410   1.7641

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.432e+00  2.716e+00  -0.527   0.5979
SocSec      -1.573e-03  7.888e-04  -1.994   0.0461 *
SocEnr       9.893e-07  5.704e-07   1.735   0.0828 .
Medicare     1.976e-03  8.948e-04   2.208   0.0273 *
SupSec      -7.242e-04  5.602e-04  -1.293   0.1961
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 68.593  on 49  degrees of freedom
Residual deviance: 53.866  on 45  degrees of freedom
AIC: 63.866

Number of Fisher Scoring iterations: 5
```

This tells me that I have a b_o=-1.432, b_1= -1.573, b_2 = 9.893, b_3= 1.976, and a b_4= -7.242.

(b)

Next, I'm going to use the predict function to output the y-hat values.

```
> glm.probs2 = predict(glm.fit2,type = "response")
> glm.probs2[1:4]
        1         2         3         4
0.14621555 0.08236176 0.49803825 0.13561615
```

Glm.probs2 is the output of estimates that will vote for president Obama. Each number on top represents the state in the csv.

```
> head(Hospital[,1],10)
 [1] AL AK AZ AR CA CO CT DE FL GA
50 Levels: AK AL AR AZ CA CO CT DE FL GA HI I
```

(c)

Now, I will take the estimated probabilities and make them into vote forecasts.

```
> glm.forcast2 = Hospital$Vote
> glm.forcast2[glm.probs>.5]="Obama"
> glm.forcast2[glm.probs<.5]="McCain"
> summary(glm.forcast)
Error in summary(glm.forcast) : objec
> summary(glm.forcast2)
McCain  Obama
    22     28
```

We take the results into a table ("Confusion Matrix") to see how many errors were made.

```
> confusion.matrix2 = table(glm.forcast2,Hospital$Vote)
> confusion.matrix2[1,1]
[1] 17
> sum(confusion.matrix2)
[1] 50
> (confusion.matrix2[1,2] + confusion.matrix2[2,1])/sum(confusion.matrix2)
[1] 0.2
```

The results show that there is a 20% error rate.

7.2 (a)

In this exercise I'm going to use logistic regression on my standardized log range data of my stock, DVN. First I created the variable Y.dvnlr to create my Y, then I made another variable glm.fit3 for my glm program.

```
> Y.dvnlr = dvnlr[,5]
> Y.dvnlr = as.factor(Y.dvnlr)
> glm.fit3 = glm(Y.dvnlr~loglag1+loglag2,data=dvnlr,family=binomial)
> summary(glm.fit3)
```

This is the result of my summary:

```
> summary(glm.fit3)

Call:
glm(formula = Y.dvnlr ~ loglag1 + loglag2, family = binomial,
    data = dvnlr)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.8232  -0.5656   0.3733   0.6871   2.2264

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.34692    0.04685   7.405 1.31e-13 ***
loglag1      4.58969    0.28552  16.075  < 2e-16 ***
loglag2      3.22728    0.27427  11.767  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 4178.5  on 3655  degrees of freedom
Residual deviance: 3132.9  on 3653  degrees of freedom
AIC: 3138.9

Number of Fisher Scoring iterations: 5
```

Looking at the data shown, it appears that my b_o= 0.34692 , b_1= 4.58969 and my b_2= 3.22728.

Now I'm going to use the predict function to output the y-hat values:

```
> glm.probs3 = predict (glm.fit3, type = "response")
> glm.probs3[1:4]
        1         2         3         4
0.9603607 0.9266043 0.8261828 0.7522599
```

The values of the output show the probabilities of the first 4 days of my data. Now using glm.forcast3, I will show the forcasts of High and Low risk days of my data.

```
> glm.forcast3 = dvnlr$Y.dvnlr
> glm.forecast3 = dvnlr$Y.dvnlr
> glm.forecast3[glm.probs >= .5] = "HighRisk"
> glm.forecast3[glm.probs<.5] = "LowRisk"
> summary(glm.forecast3)
   Length     Class      Mode
       50 character character
> table(glm.forecast3,dvnlr$Y.dvnlr)
Error in table(glm.forecast3, dvnlr$Y.dvnlr) :
  all arguments must have the same length
```

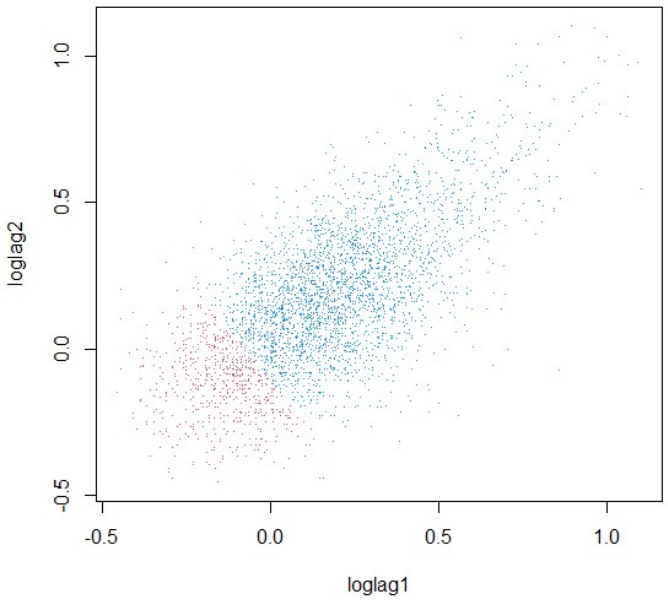
After hours of trying every way to make this forcast run, I continued to generate the same error every time.

(b)

I tried making the classification areas by using this code:

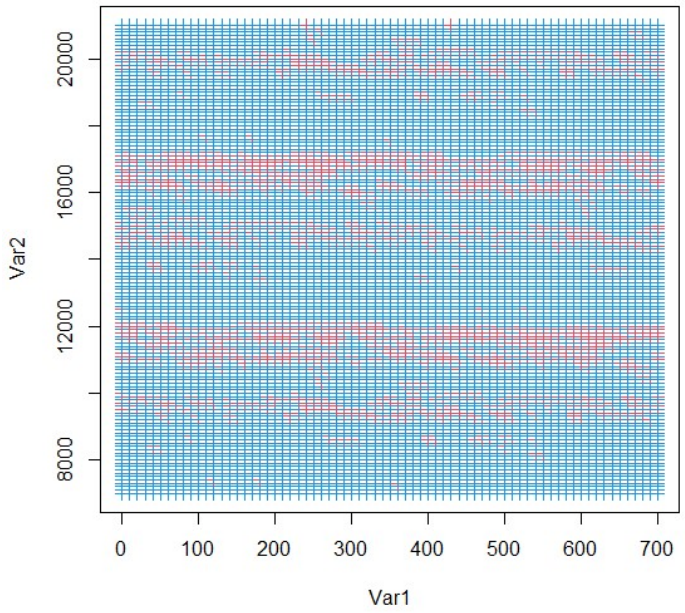
```
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
```

But it appears that I just made the points in the plot.



I tried to create a grid using the expand.grid function but ended up with scattered areas of blue and red:

```
> junk=expand.grid(seq(0,700,10),seq(7000,21000,100))
> dvn.color = 2*(1+dvn.col)
> plot(junk,col=dvn.color,pch=3)
```



The blue and red areas weren't divided as they should have. It was impossible to make the points show using:

```
> points(dvnlr[,c(3,4)],col=risk.col,pch=".")
> points(dvnlr[,c(3,4)],col=risk.col,pch=20)
> points(dvnlr[,c(3,4)],col=risk.col,pch=100)
```

I tried enlarging the points each time with the same result.

Appendix

```
ls()
library(glm)
glm
summary(glm.fit)
glm.probs[1:4]
sample(1:8,4)
coef(glm.fit)
hosp = read.csv("Hospital.csv")
history()
save.image("C:\\Users\\whall\\G
q()
load("C:\\Users\\whall\\Google
history()
history()
library(class)
Hospital = read.csv("Hospital.csv")
glm.fit2 = glm(Vote~SocSec+SocEnr+Medicare+SupSec,data=Hospital,family=binomial)
glm.fit2 = glm(Vote~Medicare+Beds,data=Hospital,family=binomial)
glm.fit2 = glm(Vote~Beds,data=Hospital,family=binomial)
glm2.fit = glm(Vote~Beds,data=Hospital,family=binomial)
load("C:\\Users\\whall\\Google Drive\\1 CUNY WORK\\0 BARUCH\\2020 Summer Baruch\'
library(class)
glm.fit2 = glm(Vote~Beds,data=Hospital,family=binomial)
glm.fit2 = glm(Vote~SocSec+SocEnr+Medicare+SupSec,data=Hospital,family=binomial)
summary(glm.fit2)
glm.probs2 = predict(glm.fit2,type = "response")
glm.probs2[1:4]
head(Hospital[,1],10)
glm.forcast2 = Hospital$Vote
glm.forcast2[glm.probs>.5]="Obama"
glm.forcast2[glm.probs<.5]="McCain"
summary(glm.forcast)
summary(glm.forcast2)
table(glm.forcast2,Hospital$Vote)
confusion.matrix2 = table(glm.forcast2,Hospital$Vote)
confusion.matrix2[1,1]
sum(confusion.matrix2)
(confusion.matrix2[1,2] + confusion.matrix2[2,1])/sum(confusion.matrix2)
history()
```

```
history()
dvnlr = read.csv("dvnlog.csv")
glm.fit3 = glm(HiLoRisk~Lag1Log+Lag2Log,data=dvnlr,family=binomial)
glm.fit2 = glm(Vote~SocSec+SocEnr+Medicare+SupSec,data=Hospital,family=binomial)
glm.fit3 = glm(HiLoRisk~Lag1Log+Lag2Log,data=dvnlr,family=binomial)
glm.fit3 = glm(HiLoRisk~Lag1Log,data=dvnlr,family=binomial)
head(dvnlr,5)
library(class)
glm.fit3 = glm(HiLoRisk~Lag1Log,data=dvnlr,family=binomial)
dim(dvnlr)
summary(dvnlr)
load("C:\\Users\\whall\\Google Drive\\1 CUNY WORK\\0 BARUCH\\2020 Summer Baruch$
load("C:\\Users\\whall\\Google Drive\\1 CUNY WORK\\0 BARUCH\\2020 Summer Baruch$
load("C:\\Users\\whall\\Google Drive\\1 CUNY WORK\\0 BARUCH\\2020 Summer Baruch$
ls()
library(class)
library(e1071)
library(caret)
library(klaR)
glm.fit3 = glm(HiLoRisk~Lag1Log,data=dvnlr,family=binomial)
glm.fit3 = glm(HiLoRisk~Lag1Log+Lag2Log,data=dvnlr,family=binomial)
head(X.dvn)
dvnlr = read.csv("dvnln6.csv")
head(dvnlr,5)
glm.fit3 = glm(dvnrisk~loglag1+loglag2,data=dvnlr,family=binomial)
glm.fit3 = glm(logdvn~loglag1+loglag2,data=dvnlr,family=binomial)
glm.fit3 = glm(dvnrisk~loglag1+loglag2,data=dvnlr,family=binomial)
Y.dvnlr = dvnlr[,5]
Y.dvnlr = as.factor(Y.dvnlr)
glm.fit3 = glm(Y.dvnlr~loglag1+loglag2,data=dvnlr,family=binomial)
summary(glm.fit3)
glm.probs3 = predict (glm.fit3, type = "response")
glm.probs3[1:4]
glm.forcast = dvnlr$Y.dvnlr
median(glm.forcast)
median(glm.probs3)
```

```
glm.forcast[glm.probs>0.8116734] = "HighRisk"
glm.forcast[glm.probs<0.8116734] = "LowRisk"
summary(glm.forecast)
glm.forcast3 = dvnlr$Y.dvnlr
median(glm.forcast3)
glm.forcast3[glm.probs>0.8116734] = "HighRisk"
glm.forcast3[glm.probs<0.8116734] = "LowRisk"
summary(glm.forecast3)
glm.forcast3 = dvnlr$Y.dvnlr
head(glm.forcast3,5)
summary(glm.forecast3)
head(dvnlr,5)
head(Y.dvnlr,5)
glm.forcast3 = dvnlr$Y.dvnlr
glm.forcast3
glm.forcast3[glm.probs>0.8116734] = "HighRisk"
glm.forcast3[glm.probs<0.8116734] = "LowRisk"
summary(glm.forecast3)
glm.forcast3 = dvnlr$dvnrisk
glm.forcast3[glm.probs>0.8116734] = "HighRisk"
glm.forcast3[glm.probs<0.8116734] = "LowRisk"
summary(glm.forecast3)
dvnlr$Y.dvnlr
table(glm.forcast3,dvnlr$Y.dvnlr)
dvnlr$Y.dvnlr
glm.forcast3
glm.forcast3[glm.probs>.5] = "HighRisk"
glm.forcast3[glm.probs<.5] = "LowRisk"
summary(glm.forecast3)
table(glm.forcast3,dvnlr$Y.dvnlr)
dim(glm.forcast3)
head(dvnlr,5)
tail(dvnlr,5)
Shuffle = sample(3656,3656)
X.dvn = dvnlr[,3:4]
LnX = log[X.dvn)
```

```
LnX = log[X.dvn)
LnX = log(X.dvn)
LnX.dvn = X.dvn
StX = apply(X.dvn,2,scale)
glm.forcast3 = StX$Y.dvnlr
glm.forcast3 = X.dvn$Y.dvnlr
table(glm.forcast3,X.dvn$Y.dvnlr)
StX = apply(LnX,2,scale)
glm.forcast3 = StX$Y.dvnlr
X.dvn = scale(X.dvn)
glm.forcast3 = X.dvn$Y.dvnlr
glm.forcast3 = X.dvn[Y.dvnlr]
glm.forcast3[glm.probs>.5] = "HighRisk"
glm.forcast3[glm.probs<.5] = "LowRisk"
summary(glm.forecast3)
summary(glm.forcast3)
glm.forcast3 = dvnlr$dvnrisk
summary(glm.forcast3)
glm.forcast3 = dvnlr$Y.dvnlr
summary(glm.forcast3)
head(Hospital)
glm.forecast3 = dvnlr$Y.dvnlr
glm.forecast3[glm.probs>.5] = "HighRisk"
glm.forecast3[glm.probs<.5] = "LowRisk"
summary(glm.forecast3)
glm.forecast3[glm.probs>.501] = "HighRisk"
glm.forecast3[glm.probs<.501] = "LowRisk"
summary(glm.forecast3)
table(glm.forecast,dvnlr$Y.dvnlr)
dvnr = read.csv("dvnrange.csv")
Y.dvn = as.factor(Y.dvn)
Y.dvn = dvnr[,5]
Y.dvn = as.factor(Y.dvn)
glm.fit4 = glm(dvnr~lag1+lag2,data=dvnr,family=binomial)
glm.fit4 = glm(dvnr~Lag1+Lag2,data=dvnr,family=binomial)
glm.fit4 = glm(Y.dvn~Lag1+Lag2,data=dvnr,family=binomial)
glm.fit4 = glm(Y.dvn~Lag 1+Lag 2,data=dvnr,family=binomial)
```

```
table(glm.forecast,dvnlr$Y.dvnlr)
summary(glm.forecast3)
contrasts(dvnlr$Y.dvnlr)
Y.dvnlr = dvnlr[,5]
Y.dvnlr = as.factor(Y.dvnlr)
contrasts(dvnlr$Y.dvnlr)
ls()
contrasts(Y.dvnlr)
glm.forecast3[glm.probs>.501] = "HighRisk"
glm.forecast3[glm.probs<.501] = "LowRisk"
summary(glm.forecast3)
table(glm.forecast3,dvnlr$Y.dvnlr)
table(glm.forecast3,Y.dvnlr)
glm.forecast3[glm.probs>=.5] = "HighRisk"
glm.forecast3[glm.probs>.5] = "HighRisk"
glm.forecast3[glm.probs=<.5] = "LowRisk"
glm.forecast3[glm.probs =<.5] = "LowRisk"
glm.forecast3[glm.probs>=.5] = "HighRisk"
glm.forecast3[glm.probs <.5] = "LowRisk"
summary(glm.forecast3)
glm.forecast3[glm.probs > .5] = "HighRisk"
glm.forecast3[glm.probs <.5] = "LowRisk"
glm.forecast3[glm.probs <.5] = "LoRisk"
summary(glm.forecast3)
table(glm.forecast3,dvnlr$Y.dvnlr)
glm.forecast3[glm.probs > .5] = "HiRisk"
glm.forecast3[glm.probs <.5] = "LoRisk"
table(glm.forecast3,dvnlr$Y.dvnlr)
class(Y.dvnlr)
history()
history(max.show=200)
save.image("C:\\Users\\whall\\Google Drive\'
q()
load("C:\\Users\\whall\\Google Drive\\1 CUN
history()
history(max.show=200)
```

```
glm.probs3 = predict (glm.fit3, type = "response")
glm.probs3 = predict (glm.fit3,X.dvn, type = "response")
X.dvn = dvnlr[,3:4]
glm.probs3 = predict (glm.fit3,X.dvn, type = "response")
length(glm.probs3)
dim(glm.probs3)
dvn.col = round(glm.probs3)
summary(dvn.col)
plot(X.dvn,col=dvn.col,pch=".")
dvn.col[1:5]
plot(dvnlr[,c(3,4)],,col=dvn.col,pch=20)
dvn.color = 2*(1+dvn.col)
dvn.color[1:5]
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=20)
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
head(dvn.col,2)
plot(X.dvn,col=dvn.color,pch".")
plot(X.dvn,col=dvn.color,pch=".")
plot(X.dvn,col=dvn.color,pch=".")
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
plot(X.dvn,col=dvn.color,pch=".")
junk=expand.grid(seq(0,700,10),seq(7000,21000,100))
plot(junk)
plot(junk,col=dvn.color,pch=".")
head(dvnlr,2)
risk = as.numeric(dvnlr$dvnrisk)
is.na(risk)
risk = as.numeric(dvnlr$Y.dvn)
head(risk)
risk[1:5]
head(Y.dvn)
risk = as.numeric(dvnlr$Y.dvnlr)
head(Y.dvnlr)
risk[1:5]
risk = as.numeric(Y.dvnlr)
risk[1:5]
risk.col = 2*risk
risk.col[1:5]
points(dvnlr[,c(3,4)],col=risk.col,pch=".")
points(dvnlr[,c(3,4)],col=risk.col,pch=20)
points(dvnlr[,c(3,4)],col=risk.col,pch=100)
points(dvnlr[,c(3,4)],col=risk.col,pch=1000)
points(dvnlr[,c(3,4)],col=risk.col,pch=500)
points(dvnlr[,c(3,4)],col=risk.col,pch=100)
junk=expand.grid(seq(0,70,1),seq(700,2100,10))
plot(junk,col=dvn.color,pch=".")
junk=expand.grid(seq(0,7,1),seq(70,210,1))
plot(junk,col=dvn.color,pch=".")
junk=expand.grid(seq(0,70,10),seq(70,210,1))
plot(junk,col=dvn.color,pch=".")
junk=expand.grid(seq(0,70,1),seq(700,2100,10))
junk=expand.grid(seq(0,700,100),seq(700,2100,10))
plot(junk,col=dvn.color,pch=".")
plot(junk,col=dvn.color,pch=20)
junk=expand.grid(seq(0,700,10),seq(7000,21000,100))
plot(junk,col=dvn.color,pch=20)
glm.probs3 = predict (glm.fit3,junk, type = "response")
glm.probs3 = predict (glm.fit3,junk, type = "response")
glm.probs3 = predict (glm.fit3,type = "response")
summary(glm.probs3)
points(dvnlr[,c(3,4)],col=risk.col,pch=100)
junk=expand.grid(seq(0,700,10),seq(7000,21000,100))
plot(junk,col=dvn.col,pch=20)
plot(junk,col=dvn.color,pch=20)
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=20)
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=5)
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=10)
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
points(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
plot(junk,col=dvn.col,pch=20)
```

```
plot(junk,col=dvn.col,pch=20)
plot(junk,col=dvn.color,pch=".")
points(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
plot(dvnlr[,c(3,4)],col=(2*(1+dvn.col)),pch=".")
plot(junk,col=dvn.color,pch=".")
plot(junk,col=dvn.color,pch=20)
plot(junk,col=dvn.color,pch=5)
plot(junk,col=dvn.color,pch=3)
> points(dvnlr[,c(3,4)],col=risk.col,pch=".")
points(dvnlr[,c(3,4)],col=risk.col,pch=".")
points(dvnlr[,c(3,4)],col=risk.col,pch=20)
points(dvnlr[,c(3,4)],col=risk.col,pch=100)
history(max.show=200)
history(max.show=200)
```