Lab Assignment 3: Quant III

Kyle Davis
Working largely alongside Andrew Goodhart

Question 1:
A new DGP:

```
set.seed(12345)
library(ggplot2)
library(MASS)
library(caret)
library(readtext) #for text data
library(quanteda) #^
library(Rlab)
library(boot)     #runs inv.logit
library(e1071)    #helps run Nbayes train()

N     <- 1000
P     <- 20
mu    <- runif(P, -1,1)
Sigma <- rWishart(n=1, df=P, Sigma=diag(P))[,,1]
Sigma <- ifelse(row(Sigma) != col(Sigma), 0, Sigma) # deletes the off diagonal
# values to ensure independence.
X     <- mvrnorm(N, mu=mu, Sigma = Sigma)
p     <- rbern(P, 0.37)
beta  <- p*rnorm(P,1,0.9) + (1-p)*rnorm(P,0,0.3)
eta   <- X%*%beta
pi    <- inv.logit(eta)
Y     <- rbern(N, pi)
#sum(Y) #half success

Y     <- as.factor(Y)
data.lab4 <- data.frame(X, Y)
```

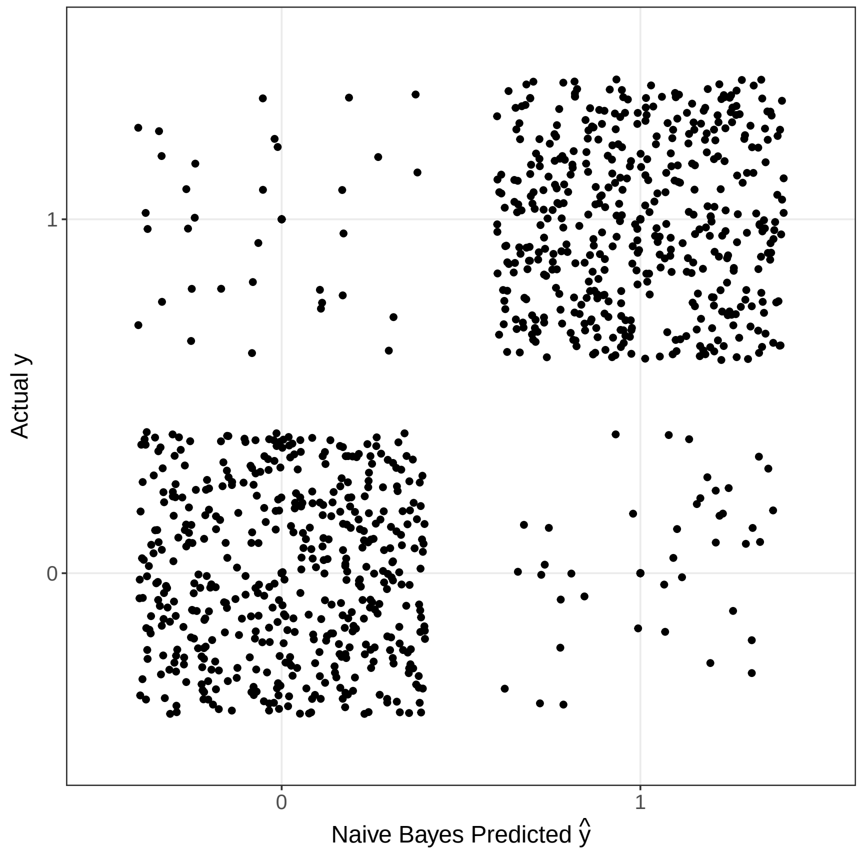
Question 2:
Let's run predictions from a Naive Bayes model and check our errors between the actual model and these predicted values.

```
model <- naiveBayes(Y ~ ., data = data.lab4)
# class(model) #Great, it reports as a naivebayes model (checking)

#Model Characteristics
# summary(model) #(checking)

#Conditional Probabilities and a-priori probabilities:
```

```r
# print(model) #these look about right, let's calculate predictions and report:

preds <- predict(model, newdata = data.lab4)
error <- as.numeric(Y) - as.numeric(preds)

# Mean Aboslute Error
mean(abs(error))

## [1] 0.073
```

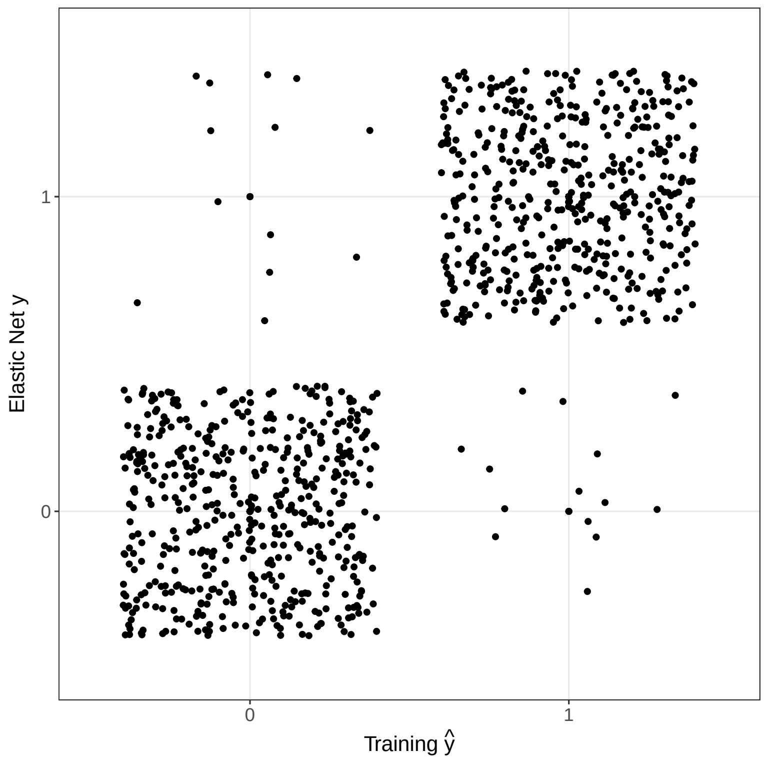
From just our errors and calculating predictions it looks like things went well (low MSE), let's plot these differences. Since I have chosen to use a dichotmous output I have jittered the final result because of overlapping.

```r
qplot(Y, preds, alpha=I(0.25))+
  geom_jitter()+
  xlab( expression(paste("Naive Bayes Predicted ", hat(y))))+
  ylab( expression(paste("Actual ", y)))+
  theme_bw()+
  theme(axis.text=element_text(size=12),
        axis.title=element_text(size=14))
```

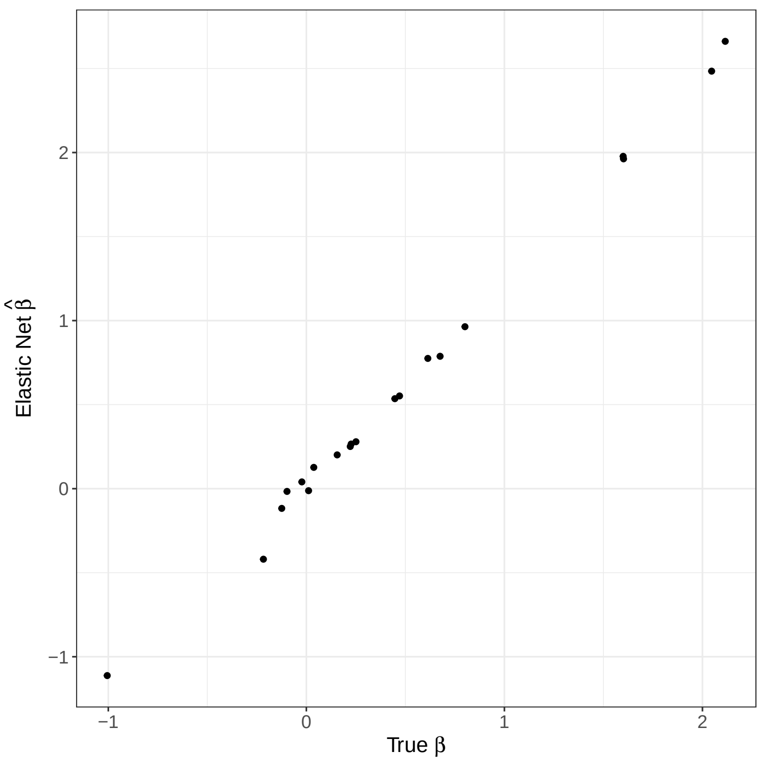


We see that a large majority of our data still remains within congruent quadrents of the graph, with some error randomly distributed to either side. Let's follow this with an elastic net for comparison:

```r
mod_enet <- train(Y~., method="glmnet",
                  tuneGrid=expand.grid(alpha=seq(0,1,0.1),
                                       lambda=seq(0,200,1)),
                  data=data.lab4,
                  preProcess=c("center"),
                  trControl=trainControl(method="cv",number=2, search="grid"))


yhat = predict(mod_enet)
qplot(Y, yhat, alpha=I(0.25))+
  geom_jitter()+
  xlab( expression(paste("Training ", hat(y))))+
  ylab( expression(paste("Elastic Net ", y)))+
  theme_bw()+
  theme(axis.text=element_text(size=12),
        axis.title=element_text(size=14))
```



```r
enet_beta = coef(mod_enet$finalModel, mod_enet$bestTune$lambda)
qplot(beta, enet_beta[-1])+
  xlab( expression(paste("True " , beta)))+
  ylab( expression(paste("Elastic Net " , hat(beta))))+
  theme_bw()+
  theme(axis.text=element_text(size=12),
        axis.title=element_text(size=14))
```



We see with the elastic net, less variation at the opposite ends. Furthermore, our beta's seem to run together pretty well. For these reasons I'll proclaim the elastic net to do a better job than the Naive Bayes but only at the margins for this case.
Question 3:

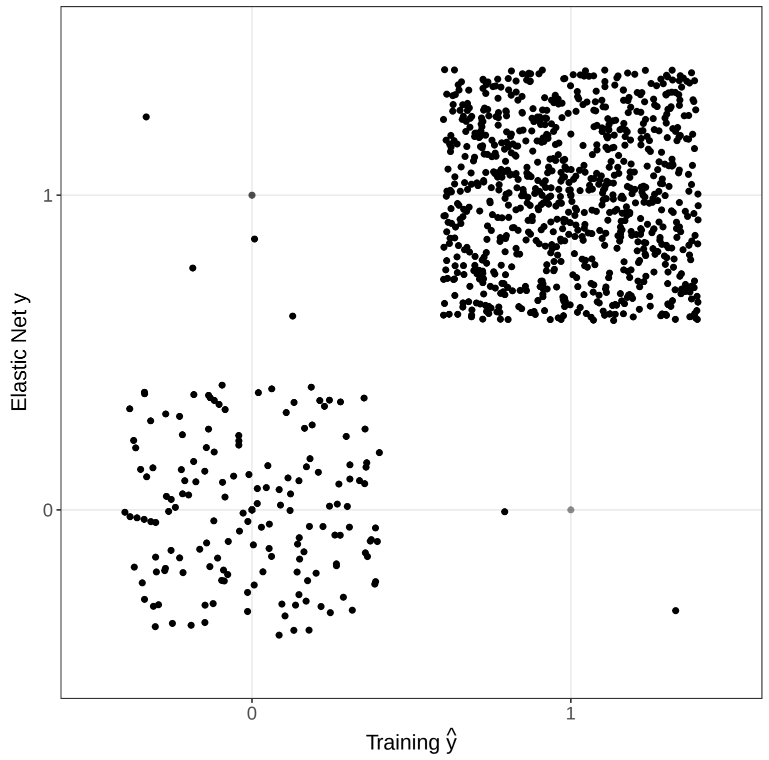
Increasing probability of success (1) and reporting results, just elastic net this time:

```r
N <- 1000
P <- 20
mu2 <- runif(P, 0.5, 1.5)
Sigma2 <- rWishart(n=1, df=P, Sigma=diag(P))[,,1]
Sigma2 <- ifelse(row(Sigma2) != col(Sigma2), 0, Sigma2)
X2 <- mvrnorm(N, mu=mu2, Sigma = Sigma2)
p2 <- rbern(P, 1)
beta2 <- p2*rnorm(P,1,0.1) + (1-p2)*rnorm(P,0,1)
eta2 <- X2%*%beta2
pi2 <- inv.logit(eta2)
Y2 <-rbern(N, pi2)
Y2 <- as.factor(Y2)
```

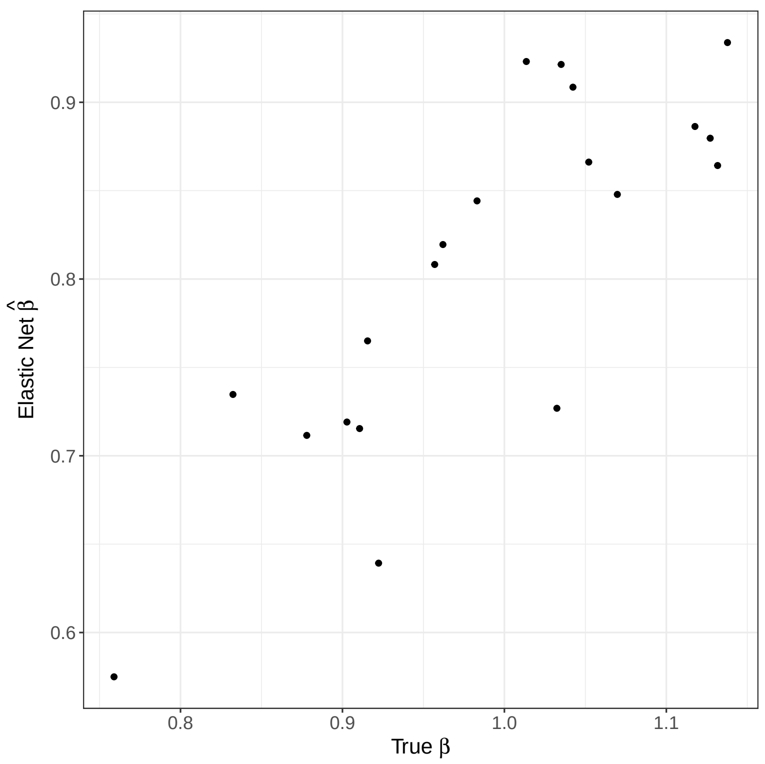
```r
data2.lab4 <- data.frame(X2, Y2)


model2 <- naiveBayes(Y2 ~ ., data = data2.lab4)
# print(model2)
preds2 <- predict(model2, newdata = data2.lab4)
error2 <- as.numeric(Y2) - as.numeric(preds2)

# Mean Aboslute Error
mean(abs(error2))
```

```
## [1] 0.07
```

```r
#Elastic Net:

mod_enet2 <- train(Y2~., method="glmnet",
                  tuneGrid=expand.grid(alpha=seq(0,1,0.1),
                                       lambda=seq(0,200,1)),
                  data=data2.lab4,
                  preProcess=c("center"),
                  trControl=trainControl(method="cv",number=2, search="grid"))


yhat2 = predict(mod_enet2)
qplot(Y2, yhat2, alpha=I(0.25))+
  geom_jitter()+
  xlab( expression(paste("Training ", hat(y))))+
  ylab( expression(paste("Elastic Net ", y)))+
  theme_bw()+
  theme(axis.text=element_text(size=12),
        axis.title=element_text(size=14))
```



```
enet_beta2 = coef(mod_enet2$finalModel, mod_enet2$bestTune$lambda)
qplot(beta2, enet_beta2[-1])+
  xlab( expression(paste("True " , beta)))+
  ylab( expression(paste("Elastic Net " , hat(beta))))+
  theme_bw()+
  theme(axis.text=element_text(size=12),
        axis.title=element_text(size=14))
```



We see that our elastic net does well fitting data even when our probabilities are more stacked at one end. Yet, our variance on our predicted Y's went down, and our betas seem to have a lot more variance. This change in variance is likely because of the lack of overlap (coverage is the official termonology I beleive) within our data. When working with less observations where our Y's are zero has lead to more variation in our betas.

Question 4:

For Question 4 I ran into multiple problems at intial stages, unfortunately I ran out of time (life got in the way this week) to troubleshoot some of these problems and come to substansive results worth noting. I'll leave some of the intial stages coding and issues here:

```
# load("file:///C:/Users/Shing/Documents/- Previous Course Info/Quant III/POLI7553_Lab4.

#Create Data Frame for use.
# sparse <- as(review_mat, "sparseMatrix")
# sparse <- as.data.frame(as.matrix(sparse))
# text.data <- data.frame(review_sentiment, sparse)
#  names(text.data)   #Check


library(stringr) #Helps clean text data

# Take out symbols and unwanted misc stuff:
#  text.data <- stringr::str_replace_all(names(text.data),"[^a-zA-Z\\s]", " ")
# Shrink down to just one remaining white space:
#  text.data <- stringr::str_replace_all(names(text.data),"[\\s]+", " ")

#  names(text.data) #check

# A way to delete X's?
# not.want <- which(names(text.data) %in% c("Var_10", "Var_2", "Var_8"))
# dat <- dat[, -not.want]
#dat
```

Beyond just learning how to use text as data, I think I missed some intutions about what the real goal in model building was in the first place. Was the goal to use tools such as LASSO to find words that were significant and then use those words in a predictive model (say, logit for binary dependent variable)? Also, we have not had much practice in how to clean data, which is surprising given it's assumed prevelance in research – how do you delete rows of data and more efficiently clean string data? I've started learning this methodology too late. I'll plan to revisit most of this for future reference.