Stat 2025- HW 3

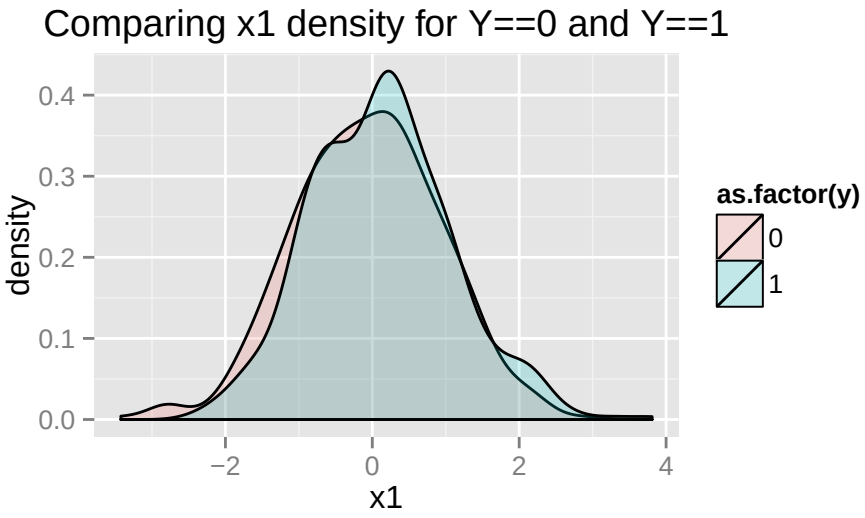
Michael Discenza

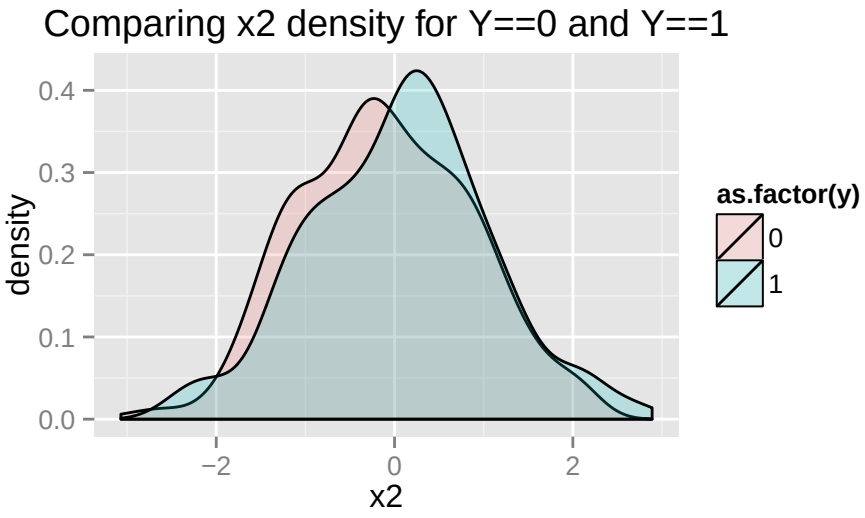
February 13, 2013

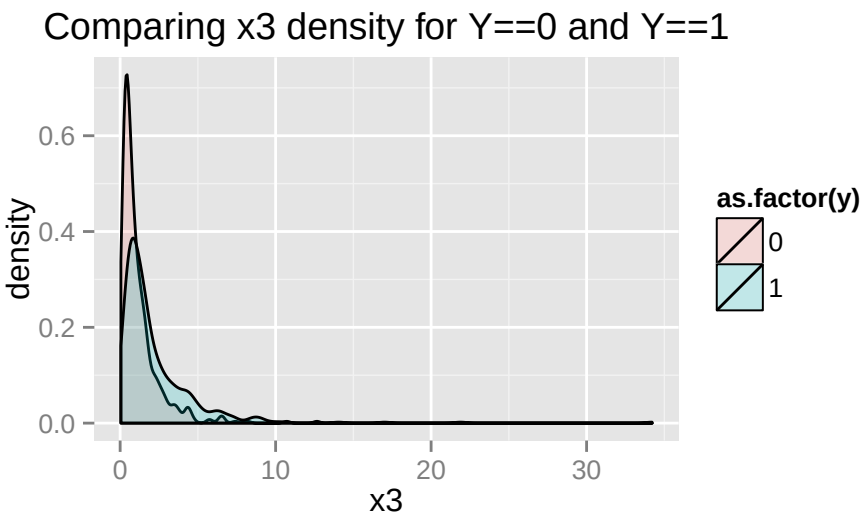
I first import the data and conduct some exploratory data analysis

```
ToyExample <- read.table("http://www.stat.columbia.edu/~madigan/W2025/data/ToyExample.dat",
    header = TRUE)
attach(ToyExample)
```

Then I plot the distribution of each predictor for responses y=0 and y=1. This provides some intuition about how the features might help us differentiate between records that are classfied by 0 and 1.



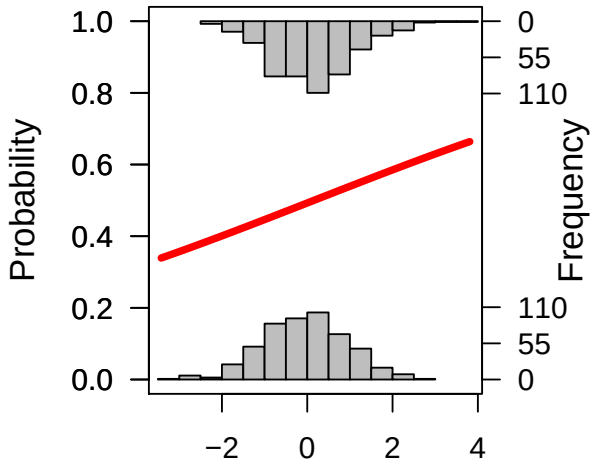




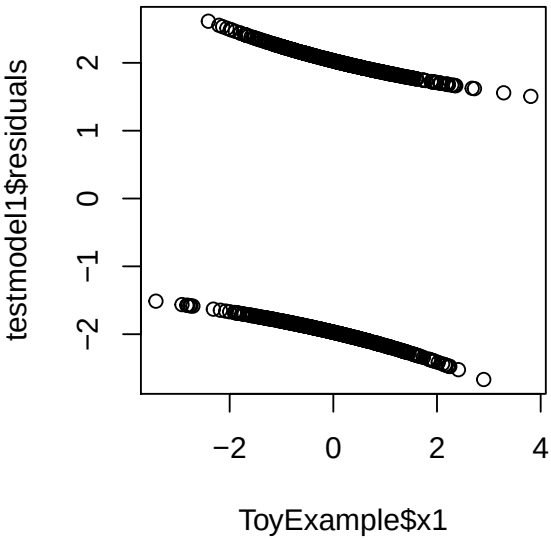
Before moving into fitting complex logistic regression models with multipl predictors, I fit models with single predictors and examine residual plots to see if there are patterns that indicate a transformation of any of the predictors might be useful. Instead of plotting the actual data points along y=0 and y=1, I plot the distributions of the x variables along those horizontal lines so that the relationships can be seen more clearly.

```
testmodel1 <- glm(y ~ x1, family = binomial, data = ToyExample)
pred1 <- predict(testmodel1, type = "response")
par(mfrow = c(1, 2))
logi.hist.plot(ToyExample$x1, ToyExample$y, boxp = FALSE, type = "hist", col = "gray", main = "Fitted Lo
plot(testmodel1$residuals ~ ToyExample$x1, main = "Residuals vs. x1")
# x2
testmodel2 <- glm(y ~ x2, family = binomial, data = ToyExample)
pred2 <- predict(testmodel2, type = "response")
par(mfrow = c(1, 2))
logi.hist.plot(x2, y, boxp = FALSE, type = "hist", col = "gray", main = "Fitted Logistic Regression with
plot(testmodel2$residuals ~ ToyExample$x2, main = "Residuals vs. x2")
# 3
testmodel3a <- glm(ToyExample$y ~ ToyExample$x3, family = binomial, data = ToyExample)
pred3 <- predict(testmodel3a, type = "response")
par(mfrow = c(1, 2))
logi.hist.plot(ToyExample$x3, ToyExample$y, boxp = FALSE, type = "hist", col = "gray")
plot(testmodel3a$residuals ~ ToyExample$x3, main = "Residuals vs. x3")
```

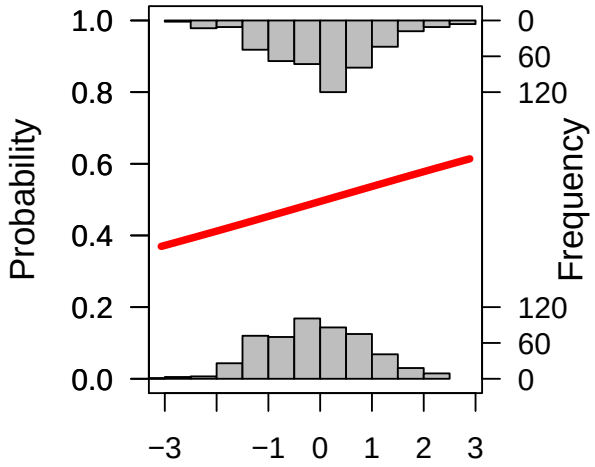
d Logistic Regression with x1 as Pred



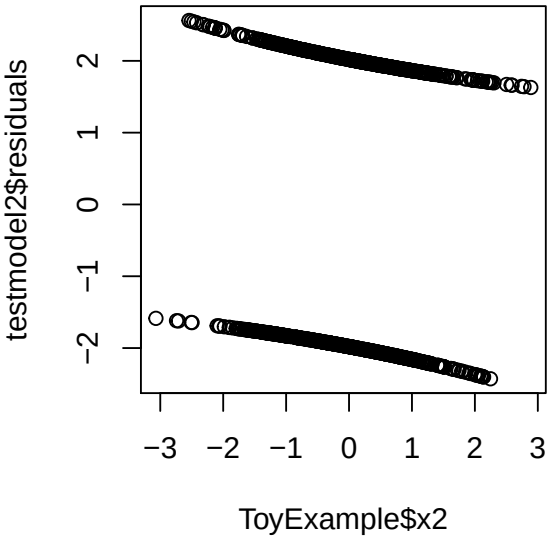
Residuals vs. x1

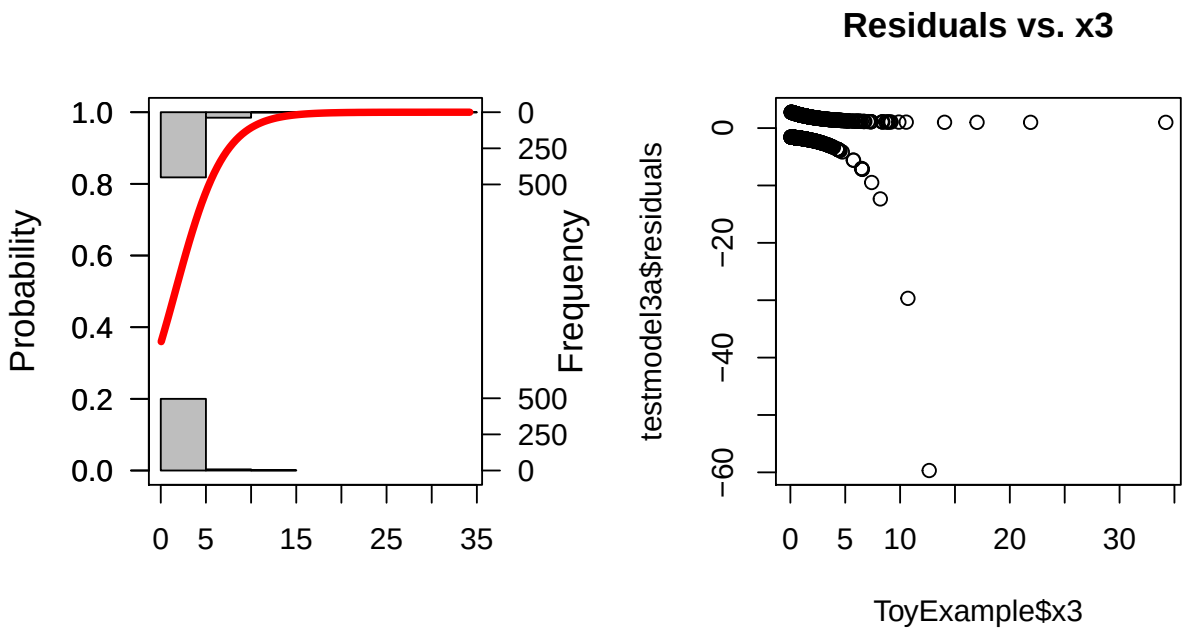


d Logistic Regression with x2 as Pred



Residuals vs. x2

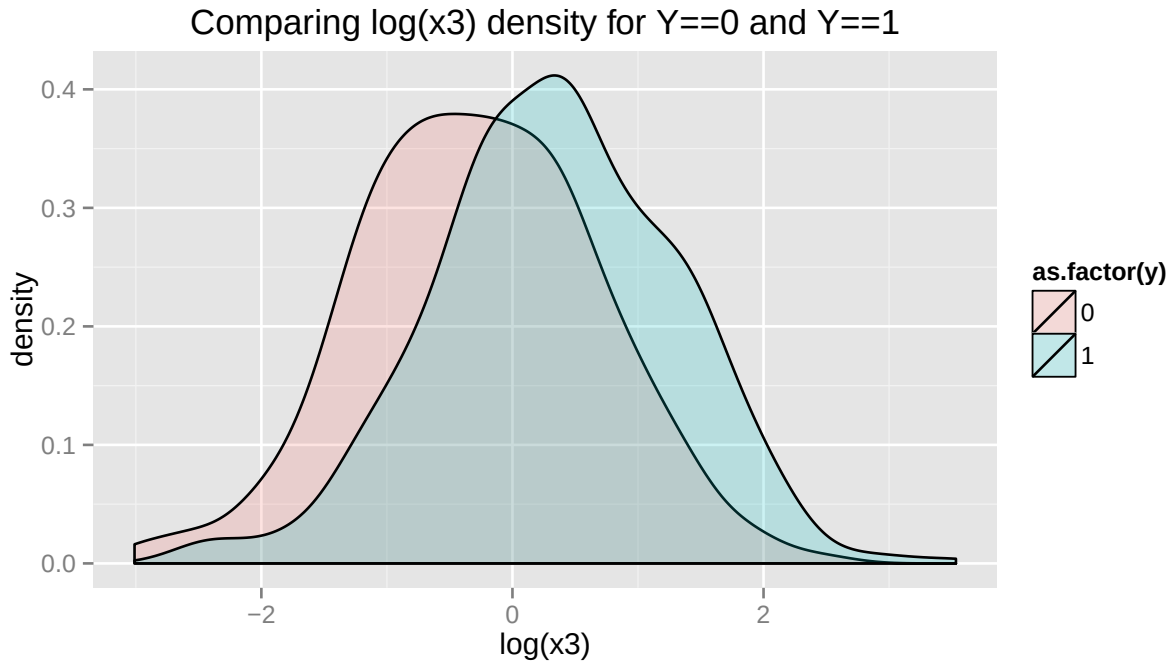


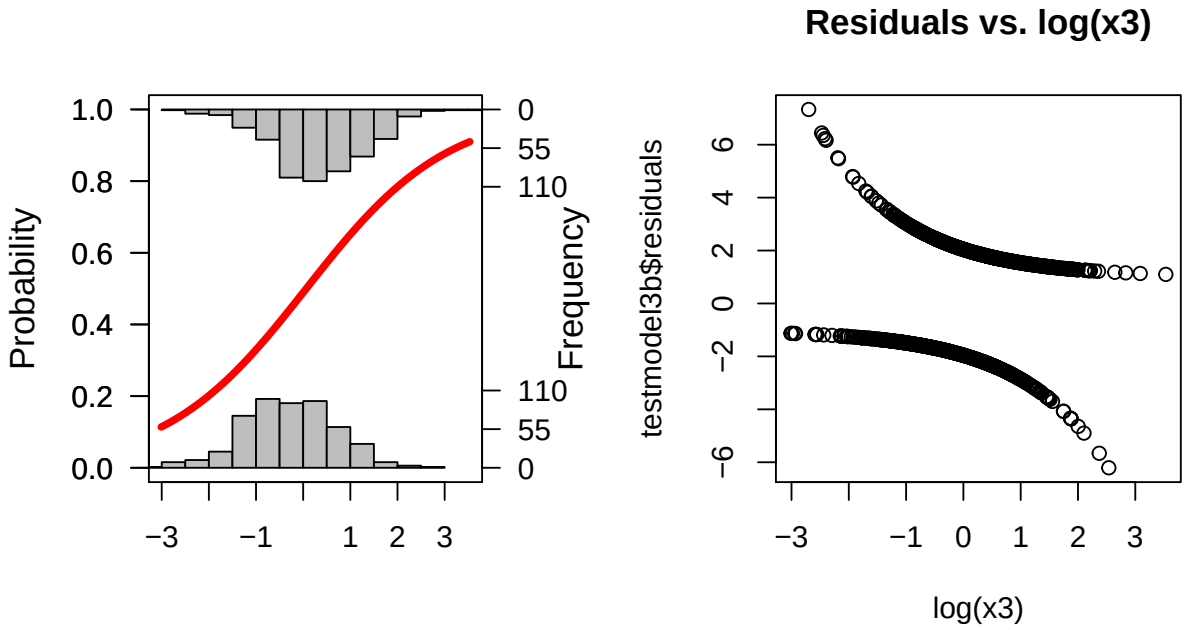


We can confirm that there are indeed issues fitting a logistic regression model to x3. To get rid of the problem with the pattern of the residuals, we transform x3 and instead fit the model to log of x3:

```
ggplot(ToyExample, aes(log(x3), fill = as.factor(y))) + geom_density(alpha = 0.2) + labs(title = expres
testmodel3b <- glm(y ~ log(x3), family = binomial, data = ToyExample)
par(mfrow = c(1, 2))
logi.hist.plot(log(x3), y, boxp = FALSE, type = "hist", col = "gray")
plot(testmodel3b$residuals ~ log(x3), main = "Residuals vs. log(x3)")

ToyExample$x3 <- log(ToyExample$x3)
colnames(ToyExample)[4] <- "logx3"
```





Having established that each of the predictors indivdually are acceptable to use in logistic regression, I then proceed to model selection by searching the model space of the combinations of x1, x2, and the log of x3. To start this procedure, I define functions to enumerate all possible models, fit theses models, and retunrn metrics for their evaluation

```
all_possible_models <- function(x.column.names, y.col.name){
    Cols <- x.column.names
    n <- length(Cols)
    id <- unlist(
```

```
        lapply(1:n,
               function(i)combn(1:n,i,simplify=F)
        )
        ,recursive=F)
    Formulas <- sapply(id,function(i)
        paste(y.col.name, "~",paste(Cols[i],collapse="+"))
    )
}
library(nlme)
apply_formula <- function(model, cv=5){ #make the default value for this function 5
    fitmodel <- glm(model, family=binomial, data=ToyExample)
    #cv.model <- CVlm(ToyExample,glm(model, family=binomial, data=ToyExample),m=cv)
    #cvmse <-mean((cv.model$sale.price - cv.model$cvpred)^2)
    AIC <- fitmodel$aic
    BIC <- BIC(fitmodel)
    result <- c(AIC, BIC)
    return (result)
}

FORMULAS <- all_possible_models(
    x.column.names=colnames(ToyExample[2:4]),
    y.col.name=colnames(ToyExample[1])
    )

FORMULAS[8] <- "y ~ 1" # add the null model
crit <- as.data.frame(lapply(X=FORMULAS, apply_formula))
names(crit) = NULL
model.selection <- cbind(FORMULAS, t(crit))
colnames(model.selection)[2] <- "AIC"
colnames(model.selection)[3] <- "BIC"
```

```
model.selection

##      FORMULAS          AIC                BIC
## [1,] "y ~ x1"          "1381.93433060897" "1391.74984116694"
## [2,] "y ~ x2"          "1383.32311957695" "1393.13863013491"
## [3,] "y ~ logx3"       "1292.62584079638" "1302.44135135434"
## [4,] "y ~ x1+x2"       "1377.93116111084" "1392.65442694779"
## [5,] "y ~ x1+logx3"    "1287.20171337768" "1301.92497921463"
## [6,] "y ~ x2+logx3"    "1291.07681023363" "1305.80007607058"
## [7,] "y ~ x1+x2+logx3" "1286.19525342673" "1305.82627454266"
## [8,] "y ~ 1"           "1388.15035766369" "1393.05811294267"
```

1 What is the best model according to AIC?

The best model is the model that includes all of the predictors, y x1+x2+logx3.

2 What is the best model according to BIC?

The best model is y x1+logx3, which makes sense because tends to chose smaller models. e

3 What is the best model using lasso?

```
library(glmnet)
glmnetModel <- glmnet(x = as.matrix(ToyExample[, 2:4]), y = ToyExample[, 1], family = "binomial")
# plot(glmnetModel)
my.cv <- cv.glmnet(x = as.matrix(ToyExample[, 2:4]), y = ToyExample[, 1], family = "binomial")
predict(glmnetModel, type = "coef", s = my.cv$lambda.min)  # returns the coefficents for the best model

## 4 x 1 sparse Matrix of class "dgCMatrix"
##                      1
## (Intercept) -0.04504
## x1           0.16969
## x2           0.10837
## logx3        0.64942
```

Here we see that the best model fit by lasso is one that includes x1, x2, and log(x3) and they are indeed smaller than the coefficeints that are fit by the regular logistic regression model fit to the same variables because of the shrinkage effect of lasso.

```
glm(y ~ x1 + x2 + logx3, data = ToyExample, family = "binomial")$coefficients

## (Intercept)          x1          x2       logx3
##    -0.04552     0.17923     0.11696     0.66050
```

4 Are there any transformations of the xs that appear to help? Explain.

We just calculate the BIC and AIC for all of the models that would be fit on the non-log transformed x3 variable and note that these models have higher AICs and BICs, indicating that the transformed models are more accurate.

```
comp

##      FORMULAS          AIC-transformation BIC-transformation AIC
## [1,] "y ~ x1"          "1381.93433060897" "1391.74984116694" "1381.93433060897"
## [2,] "y ~ x2"          "1383.32311957695" "1393.13863013491" "1383.32311957695"
## [3,] "y ~ logx3"       "1292.62584079638" "1302.44135135434" "1313.93923432423"
## [4,] "y ~ x1+x2"       "1377.93116111084" "1392.65442694779" "1377.93116111084"
## [5,] "y ~ x1+logx3"    "1287.20171337768" "1301.92497921463" "1308.48145432214"
## [6,] "y ~ x2+logx3"    "1291.07681023363" "1305.80007607058" "1311.76294856812"
## [7,] "y ~ x1+x2+logx3" "1286.19525342673" "1305.82627454266" "1306.94282624749"
## [8,] "y ~ 1"           "1388.15035766369" "1393.05811294267" "1388.15035766369"
##      BIC
## [1,] "1391.74984116694"
## [2,] "1393.13863013491"
## [3,] "1323.7547448822"
## [4,] "1392.65442694779"
## [5,] "1323.20472015908"
## [6,] "1326.48621440507"
## [7,] "1326.57384736342"
## [8,] "1393.05811294267"
```