2550 Problem Set 4

Alyson Singleton (collaborated with Katherine Webb)

11/1/2019

Problem Statement

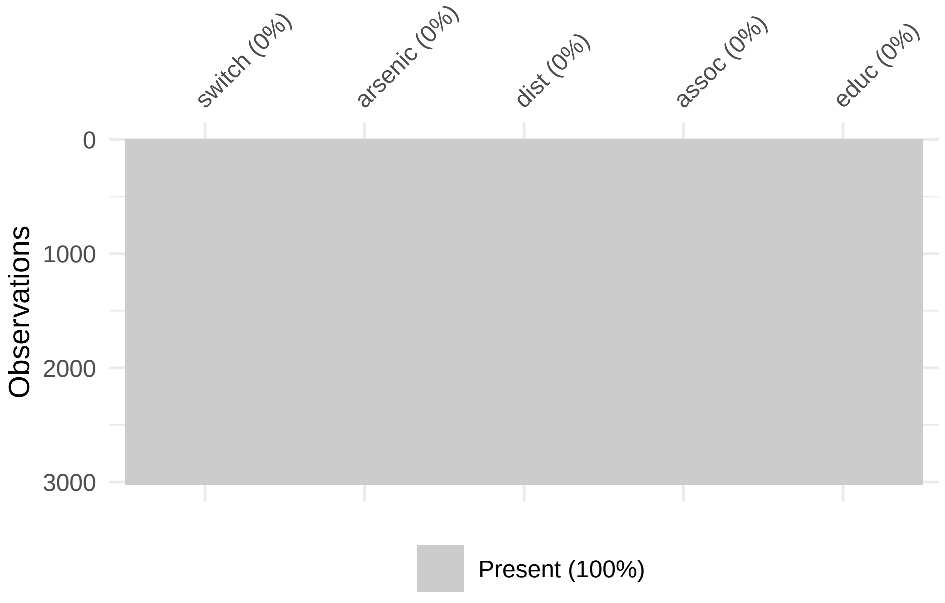
1. Wells in Bangladesh used for drinking are often contaminated by natural arsenic which can cause diseases as exposure accumulates in someone's body. If someone's well is contaminated, a neighbor's well may be safe and so if they agree, one can switch to sharing their well. After a research team measured arsenic levels in all the wells in a certain area, residents were urged to switch wells if theirs was labelled unsafe (more than 0.5 hundred micrograms per liter, i.e. 50 micrograms). A few years later the researchers returned to see who had switched wells. The data are in the file wells.txt, found in the data sets folder. They include 5 variables for 3020 wells

- switch is a binary indicator for whether the household switched wells
- arsenic is the level of arsenic in the well in hundreds of micrograms per liter
- dist is the distance to the nearest safe well in meters
- assoc is whether household members are active in community organizations
- educ is the number of years of education of the head of household

First construct a training data set of the first 2520 wells and a test data set of the last 500. I make a note here that I do not necessarily believe this is the best method of spliting the data into testing and training sets. As we are directed to do it this way I leave it for this problem set, but, as discussed in class, there are many options of how to make this decision.

Data Exploration

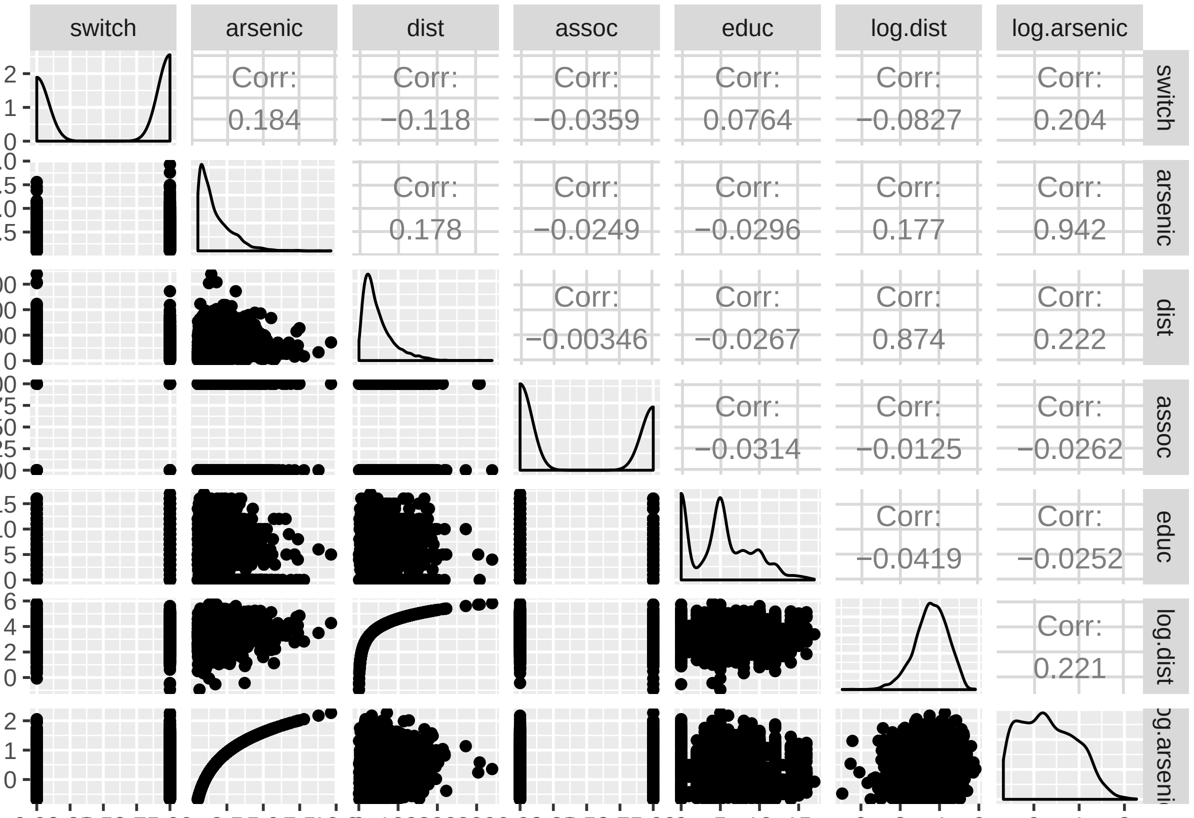
There is no missing data and therefore no need to make adjustments in that regard. See below.



Here we take a specific look into the categorical variables and see that there are plenty of samples in each of the categories.

- Switch: a binary indicator for whether the household switched wells (0 - 1283, 1 - 1737)
- Assoc: assoc is whether household members are active in community organizations (0 - 1743, 1 - 1277)

We also use a pairs plot to determine that there are no substantial correlations between any of our variables, we therefore keep all for the subsequent parts of our analysis. We also use these plots to look at the continuous variables. We notice the right skew in the spreads of arsenic and distance and decide log transforms are appropriate. The log transforms of arsenic and distance are included in the pairs plot for reference.



Logistic Regression

Part A

Construct a good logistic regression model predicting the decision to switch wells as a function of the 4 predictors (arsenic, distance, association and education) on the training data. Consider potential transformations of continuous variables and possible interactions.

Model Selection

I started by investigating a baseline model with all four exposure variables with distance and arsenic log transformed, as discussed above. Then I looked at a model with all of the two by two interaction terms. I found that a few of the interaction terms were significant, and the ANOVA Chi-Squared test showed that a model including these terms was substantically different with a lower AIC that then baseline model. I used the Chi-Squared ANOVA test because we are trying to predict a binary outcome. However, I believe that this model is fitting mostly to excess noise, not adding additional prediction strength atop that of the baseline model. With this in mind, I tried a model that only keeps the interaction terms that seemed logically viable. This resulted in only including the interaction term between distance and association (the more central you are to the community might influence the effect of how far away you are influencing your probability to switch). However, adding this interaction by itself makes no significant difference, again using the Chi-Squared ANOVA test. Therefore, we leave the baseline model as our model of choice.

Model Summary

My final model proposal is as follows :

switch ~ log.arsenic + log.dist + assoc + educ, with a logit link function (logistic regression model).

Model Output:

Table 1: Logistic Regression Summary

	Estimate	Std. Error	P-Value
Intercept	1.31805	0.20095	5.41e-11
log(Arsenic)	0.82802	0.07534	< 2e-16
log(Distance)	-0.35952	0.05276	9.45e-12
Association	-0.16912	0.08455	0.045478
Education	0.041040	0.010590	0.000106

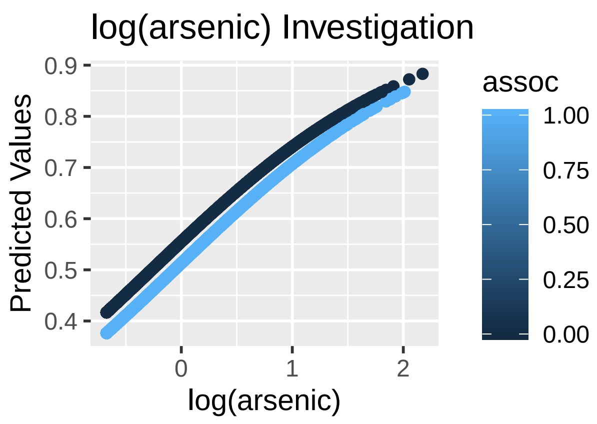
Interpretation of Coefficients

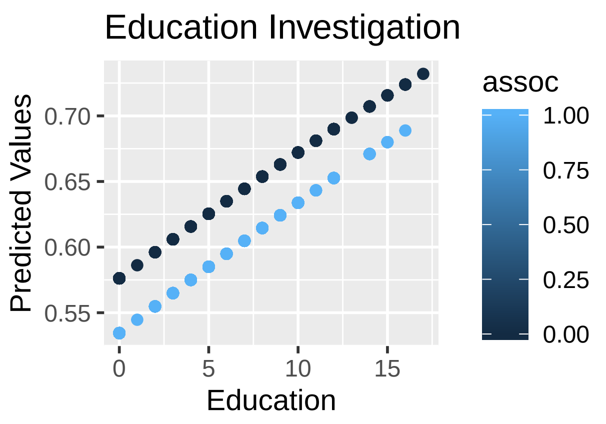
We conclude that all of the variables are predictive factors of whether or not someone switches wells. I would interpret the results of this model by saying that the logs odds of someone switching wells when all other variables are zero is 1.32. Additionally, 0.83 is the association of log(arsenic) and the log odds of switching wells while keeping all other predictors the same. We can exponentiate this to say that 2.29 is the association of arsenic and the log odds of switching wells while keeping all other predictors 0. This makes sense, as we would expect the odds of switching to increase as the arsenic levels increase. We may interpret the other variables in this same way, exponentiating when appropriate.

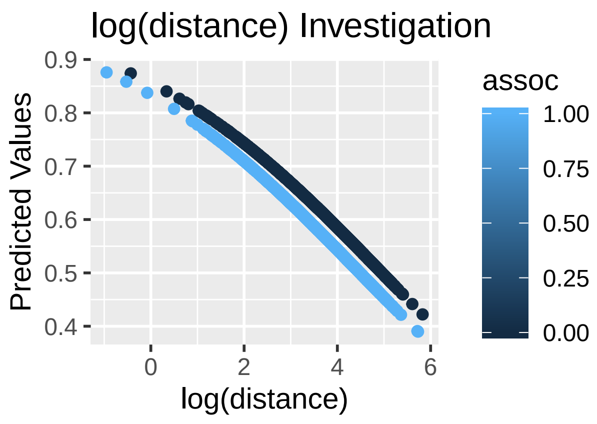
Part B

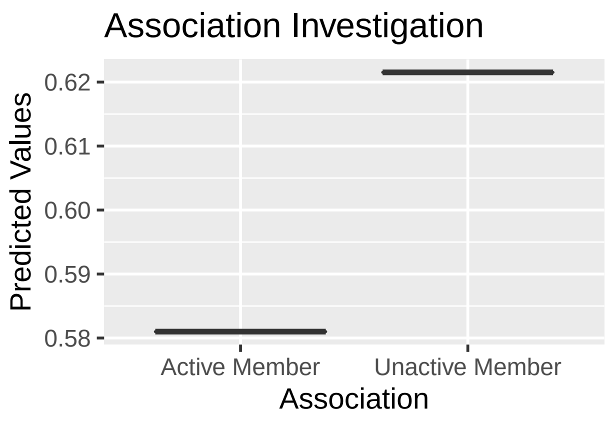
Compute and graph the predicted probabilities stratifying by each of the predictors *while holding all other predictors constant.* You could do this using graphs such as in the papers we discussed in class or by using contour plots which would allow you to graph two continuous predictors on the same plot. You can array different lines and plots to try to put this all on one sheet or you can spread across different plots. See what works best.

Below you can see a multiplot of each of these predictive plots. Each plot investigates the effects of the indicated variable while holding all others constant (using the mean values of the continuous variables). I thought it would be more accurate to show the difference between the binary classification / variable association rather than average it, that is why you can see the two different lines in all plots but the plot that investigated association itself. All show trends interesting trends! We see that as arsenic levels go up, so does the probability of switching. Similarly, as education level goes up, so does the probability of switching wells. On the other hand, as distance from the nearest safe well increases, the probability of switching decreases. Lastly, we see that being an active member actually seems to be associated with a lower probability of switching! Very interesting, and worth noting. This is also reflected with the sign of the corresponding coefficient in the above regression model.









Part C

Compute the confusion matrix on the test data using p = 0.5 as a cutoff and discuss what this tells you about the predictive model you have constructed (e.g. sensitivity, specificity, error rate, etc.)

Using the confusion matrix, we calculate that the sensitivity is 0.8376 and the specificity is 0.3045. Lastly, we calculate the error rate as 45%.

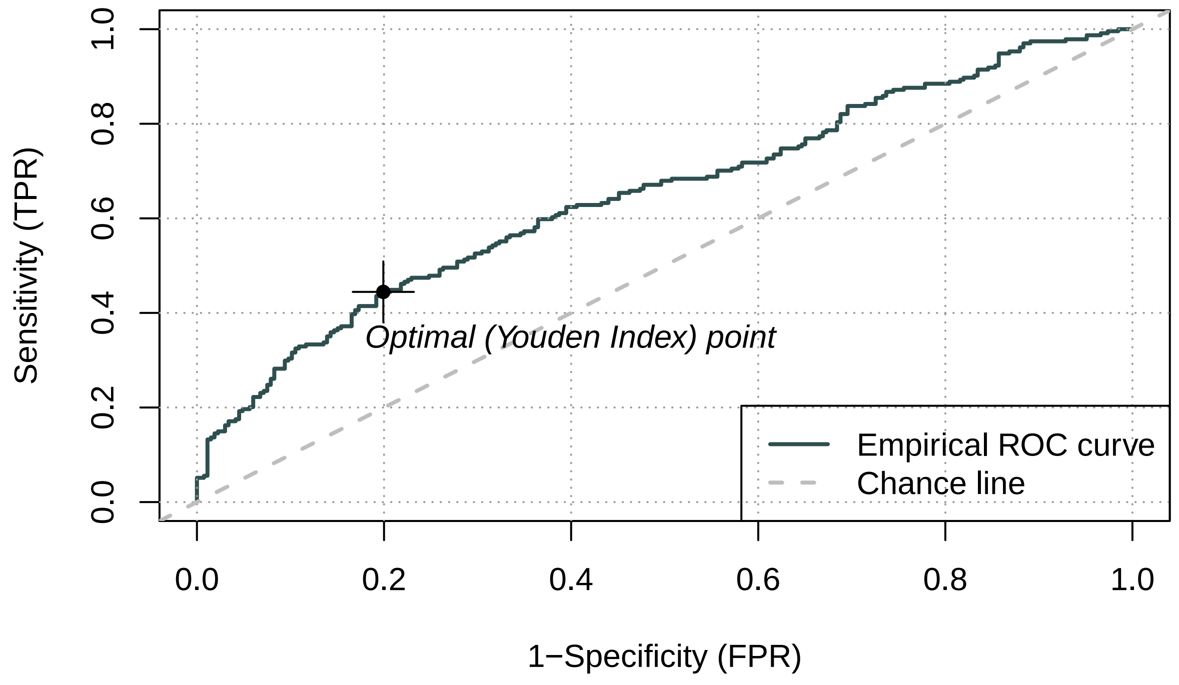
Table 2: Logistic Regression Confusion Matrix

	0	1
0	81	38
1	185	196

Part D

Construct an ROC plot and compute the area under the ROC curve.

See the ROC plot below. The area under the ROC curve is 0.6498.



Part E

What does this curve tell you about choice of threshold that balances sensitivity with specificity (i.e., how would you balance risk of switching and not switching?)

From this plot it is unclear if changing the threshold will improve our model, given there is no natural shift in the ROC curve. We note that the plot below actually shows the optimal point (thank you ROCit package). However, this does not seem like a substantial conclusion given that the curve generally has a similar shape across all threshold values. This indicates that there is not much to be improved upon by shifted the threshold value.

Part F

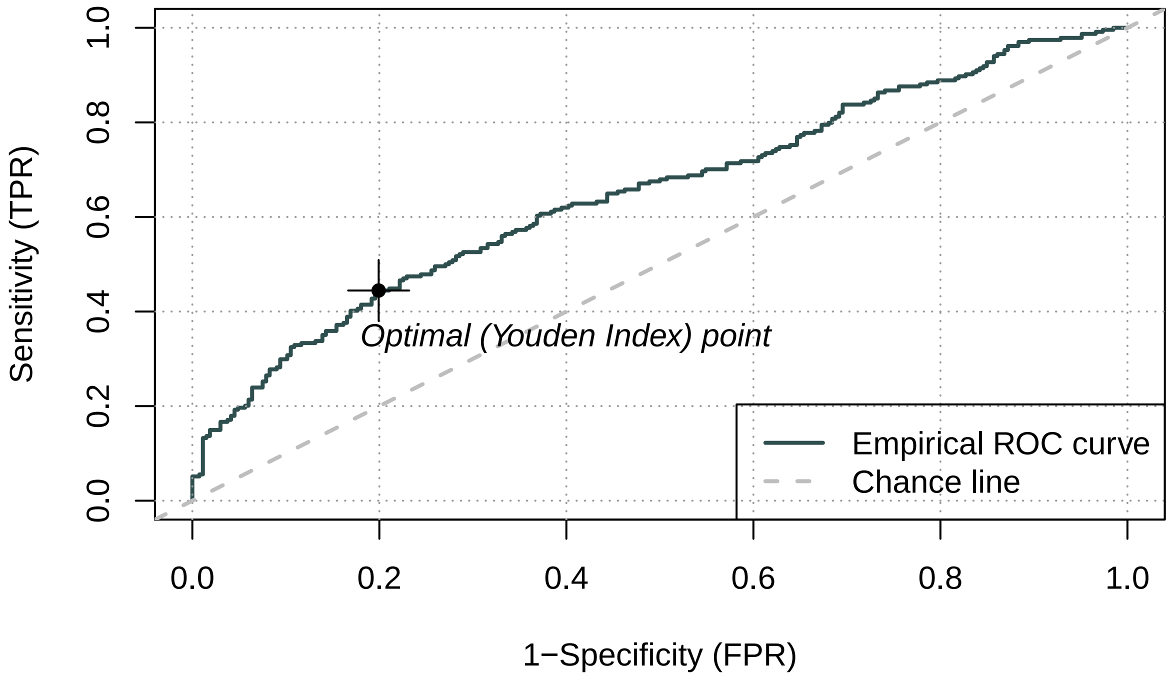
Repeat this analysis using linear discriminant analysis, quadratic discriminant analysis and K nearest neighbor with K = 1 and K = 5. For discriminant analysis, note that the predict function returns 3 elements: class is a binary indicator as to whether the posterior probability is greater than 0.5, posterior gives the posterior predictive probability and x contains the linear discriminant for the LDA function (missing for QDA). Note that for KNN, you will only get classifications, not probabilities.

Linear Discriminant Analysis

I decided to keep all variables the same for the following analyses except to remove association (assoc). We identified it as less significant in the logistic model selection above and it is binary. This works out very nicely, as we ideally only want to be using continuous predictors in LDA, QDA, and KNN analyses. Therefore, I have removed it for the following sections. Using the confusion matrix, we calculate that the sensitivity is 0.8376 and the specificity is 0.3007.

Table 3: Linear Discriminant Analysis Confusion Matrix

	0	1
0	80	38
1	186	196

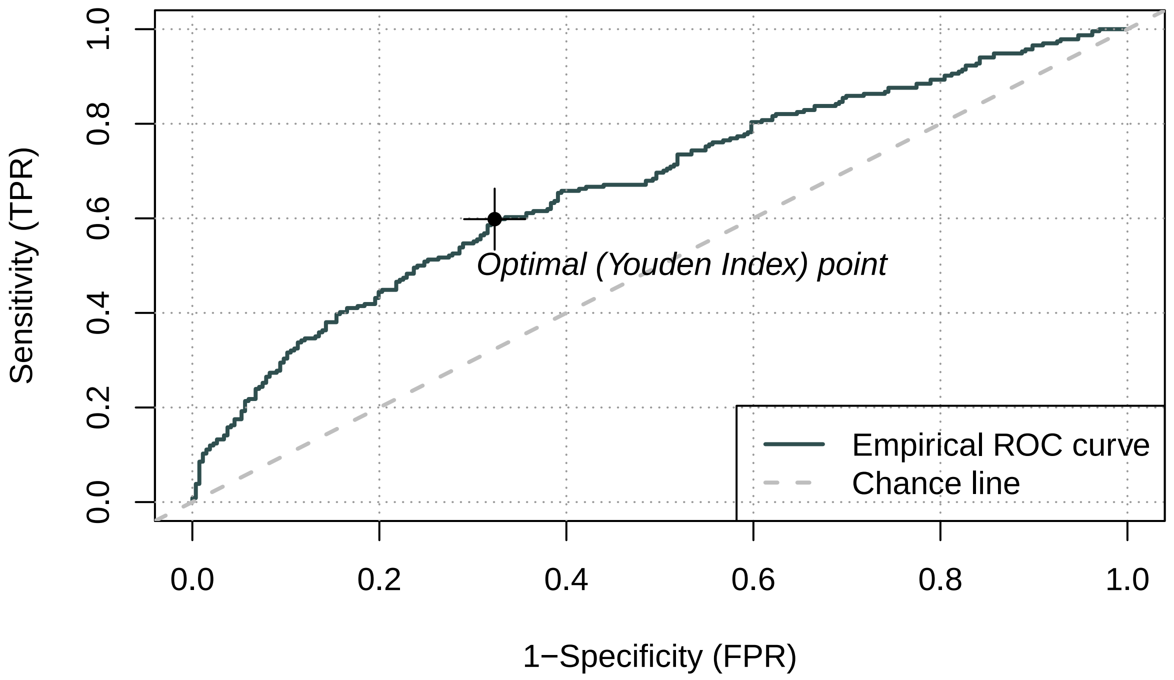


Quadratic Discriminant Analysis

Using the confusion matrix, we calculate that the sensitivity is 0.8205 and the specificity is 0.3571. Lastly, we calculate the error rate as 42.6%. See the ROC plot below. The area under the ROC curve is 0.6696.

Table 4: Quadratic Discriminant Analysis Confusion Matrix

	0	1
0	95	42
1	171	192



K Nearest Neighbor with K = 1 and K=5.

Using the confusion matrix when K=1, we calculate that the sensitivity is 0.6196 and the specificity is 0.4511. We also calculate the error rate as 47%. We do the same calculations but setting K=5. The sensitivity is 0.7094 and specificity is 0.3947. Error rate is lower at 45.8%. See both matrices below.

Table 5: KNN w K=1 Confusion Matrix

	0	1
0	120	89
1	146	145

Table 6: KNN w K=5 Confusion Matrix

	0	1
0	105	68
1	161	166

Comparison of all models' classification

Unfortunately, none of the models instill much confidence. Generally, we can see that as we increase model complexity the error rate decreases. I have listed the ranking of models in the right-most column in the comparison table below based on error rate. The best model, based on error rate, is the QDA model. LDA is next best, over both of the nearest neighbors analysis and logistic regression. Logistic regression takes third by preforming better than the KNN models. Lastly, KNN when k=5 preforms better than when k=1, as we would expect. AUC values follow these trends for the models for which we can calculate it. Sensitivity and specificity values generally follow the trends as well, although vary a bit. I would recommend the QDA out of the five shown here, although I would condition that recommendation on the fact that we might need to continue our research / try to investigate other predictive variables.

Table 7: Model Performance Comparison

	Sensitivity	Specificity	Error Rate	AUC	My Ranking
Logistic Regression	0.8376	0.3045	45%	0.6498	3
Linear Discriminant	0.8376	0.3007	44.8%	0.6505	2
Quadratic Discriminant	0.8205	0.3571	42.6%	0.6696	1
KNN w/ k=1	0.6196	0.4511	47%	NA	5
KNN w/ k=5	0.7094	0.3947	45.8%	NA	4

Code Appendix

```r
knitr::opts_chunk$set(echo = FALSE, results='asis', warning=FALSE, message = FALSE, cache=F)
setwd("~/Desktop/masters/PHP2550/pset4")
library(dplyr)
library(tidyr)
library(reshape2)

#read in data set
wells <- read.csv("wells.txt", sep="")

#separate into training and testing data
training.wells <- wells[1:2520,]
testing.wells <- wells[2521:3020,]
# missing values investigation
library(naniar)
vis_miss(wells)
# categorial variables counts
#table(wells$switch)
#table(wells$assoc)

# table(wells$educ)

# build log transforms of arcenic and distance

wells <- wells %>%
  mutate(log.dist = log(dist),
         log.arsenic = log(arsenic))
training.wells <- wells[1:2520,]
testing.wells <- wells[2521:3020,]

# pairs plots

library(GGally)
ggpairs(wells,
    diag=list(continuous="density", discrete="bar"), axisLabels="show")
```

```r
logistic.model.baseline <- glm(switch ~ log.arsenic + log.dist + assoc + educ, family = binomial, data =
#summary(logistic.model.baseline)

# all the two way interactions
logistic.model.ints <- glm(switch ~ (log.arsenic + log.dist + assoc + educ)^2, family = binomial(link =
#summary(logistic.model.ints)

# all the two way interactions
logistic.model.oneint <- glm(switch ~ log.arsenic + (log.dist * assoc) + educ, family = binomial(link =
#summary(logistic.model.oneint)

#anova(logistic.model.baseline, logistic.model.ints, test="Chisq")
#anova(logistic.model.baseline, logistic.model.oneint, test="Chisq")
library(knitr)
library(kableExtra)
library(ggplot2)

logistic.model.baseline.summ.df <- data.frame(c(1.31805, 0.20095, "5.41e-11"),
                                              c(0.82802, 0.07534, "< 2e-16"),
                                              c(-0.35952, 0.05276, "9.45e-12"),
                                              c(-0.16912, 0.08455, "0.045478"),
                                              c(0.04104, 0.01059, 0.000106))
logistic.model.baseline.summ.df <- t(logistic.model.baseline.summ.df)
colnames(logistic.model.baseline.summ.df) <- c("Estimate", "Std. Error", "P-Value")
rownames(logistic.model.baseline.summ.df) <- c("Intercept",
                                               "log(Arsenic)",
                                               "log(Distance)",
                                               "Association",
                                               "Education")

#build table
kable(logistic.model.baseline.summ.df, "latex", caption = "Logistic Regression Summary", booktabs = T) %
kable_styling(latex_options = c("striped", "hold_position"))

#mcalindon.long.nona$predictions <- predict(multi.mod.mcalindon.time.bmi.opiate)

#ggplot(mcalindon.long.nona) +
#  geom_bar(aes(x=switch, y=predictions)) +
#  labs(title="Empiric Values v. Predicted Values", x="Empiric Switch Score", y="Model Predictions")


library(ggplot2)

# Combine the hypothetical data and predicted values
# already in training wells

# Calculate confidence intervals
#std <- qnorm(0.95 / 2 + 0.5)
#new.data$ymin <- model$family$linkinv(new.data$fit - std * new.data$se)
#new.data$ymax <- model$family$linkinv(new.data$fit + std * new.data$se)
#training.wells$fit <- logistic.model.baseline$family$linkinv(training.wells$fit)  # Rescale to 0-1
```

```
# Variable : Log(arsenic)

# create data set holding all other variables constant
investigate.logarsenic.df <- training.wells
investigate.logarsenic.df$arsenic <- NULL
investigate.logarsenic.df$dist <- NULL
investigate.logarsenic.df$educ <- mean(investigate.logarsenic.df$educ)
investigate.logarsenic.df$log.dist <- mean(investigate.logarsenic.df$log.dist)

investigate.logarsenic.df$predictions <- predict(logistic.model.baseline, investigate.logarsenic.df, ty

#plot data
p1 <- ggplot(investigate.logarsenic.df, aes(x=log.arsenic, y=predictions)) +
  geom_point(aes(colour=assoc)) +
  #geom_ribbon(data=training.wells, aes(y=fit, ymin=ymin, ymax=ymax), alpha=0.5) +
  #stat_smooth(data=investigate.logarsenic.df, aes(y=predictions)) +
  labs(x="log(arsenic)", y="Predicted Values", title="log(arsenic) Investigation")


# Variable : Log(distance)

# create data set holding all other variables constant
investigate.logdistance.df <- training.wells
investigate.logdistance.df$arsenic <- NULL
investigate.logdistance.df$dist <- NULL
investigate.logdistance.df$educ <- mean(investigate.logdistance.df$educ)
investigate.logdistance.df$log.arsenic <- mean(investigate.logdistance.df$log.arsenic)

investigate.logdistance.df$predictions <- predict(logistic.model.baseline, investigate.logdistance.df, 

#plot data
p2 <- ggplot(investigate.logdistance.df, aes(x=log.dist, y=predictions)) +
  geom_point(aes(colour=assoc)) +
  #geom_ribbon(data=investigate.logdistance.df, aes(y=fit, ymin=ymin, ymax=ymax), alpha=0.5) +
  #stat_smooth(data=investigate.logdistance.df, aes(y=predictions)) +
  labs(x="log(distance)", y="Predicted Values", title="log(distance) Investigation")


# Variable : educ

# create data set holding all other variables constant
investigate.educ.df <- training.wells
investigate.educ.df$arsenic <- NULL
investigate.educ.df$dist <- NULL
investigate.educ.df$log.dist <- mean(investigate.educ.df$log.dist)
investigate.educ.df$log.arsenic <- mean(investigate.educ.df$log.arsenic)

investigate.educ.df$predictions <- predict(logistic.model.baseline, investigate.educ.df, type = "respons

#plot data
p3 <- ggplot(investigate.educ.df, aes(x=educ, y=predictions)) +
  geom_point(aes(colour=assoc)) +
  #geom_ribbon(data=investigate.educ.df, aes(y=fit, ymin=ymin, ymax=ymax), alpha=0.5) +
  #stat_smooth(data=investigate.educ.df, aes(y=predictions)) +
  labs(x="Education", y="Predicted Values", title="Education Investigation")


# Variable : assoc

# create data set holding all other variables constant
investigate.assoc.df <- training.wells
investigate.assoc.df$arsenic <- NULL
investigate.assoc.df$dist <- NULL
investigate.assoc.df$log.dist <- mean(investigate.assoc.df$log.dist)
investigate.assoc.df$log.arsenic <- mean(investigate.assoc.df$log.arsenic)
investigate.assoc.df$educ <- mean(investigate.assoc.df$educ)

investigate.assoc.df$predictions <- predict(logistic.model.baseline, investigate.assoc.df, type = "respo
investigate.assoc.df$assoc = as.factor(ifelse(investigate.assoc.df$assoc==1, "Active Member", "Unactive

#plot data
p4 <- ggplot(investigate.assoc.df, aes(x=assoc, y=predictions)) +
  geom_boxplot() +
  #geom_ribbon(data=investigate.assoc.df, aes(y=fit, ymin=ymin, ymax=ymax), alpha=0.5) +
  #stat_smooth(data=investigate.assoc.df, aes(y=predictions)) +
  labs(x="Association", y="Predicted Values", title="Association Investigation")

#multiplot function from online! cookbook-r : http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_p
multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
  library(grid)

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)

  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                     ncol = cols, nrow = ceiling(numPlots/cols))
  }

 if (numPlots==1) {
    print(plots[[1]])

  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}

multiplot(p1, p2, p3, p4, cols=2)
training.predictions <- predict(logistic.model.baseline, type = "response")
testing.predictions <- predict(logistic.model.baseline, testing.wells, type = "response")

glm.testing.pred=rep(0,500)
glm.testing.pred[testing.predictions>.5]=1 #Predict event if p > 0.5
t <- table(glm.testing.pred,testing.wells$switch) #Predicted(column) vs. observed(row)
con.table.df <- t(data.frame(t[1,],t[2,]))
rownames(con.table.df) <- c(0,1)
#con.table.df
#mean(glm.testing.pred==training.wells$switch,na.rm=T)  #correct predictions
error.rate <- 1-(mean(glm.testing.pred==training.wells$switch,na.rm=T))
#error.rate
#table(kidney$STUDY)
#train=(kidney$STUDY != 11) # training set uses data from studies 1-10
#test=kidney[!train,] #Study 11 is test data

library(knitr)
library(kableExtra)
library(dplyr)

kable(con.table.df, "latex", caption = "Logistic Regression Confusion Matrix", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
library(ROCit)

ROCit_obj <- rocit(score=testing.predictions,class=testing.wells$switch)
plot(ROCit_obj)
#summary(ROCit_obj)
library(MASS)

## Additionally, we should keep the log transforms because they help us meet the underlying assumption

# same model as above, to ask about variable selection
lda.baseline=lda(switch ~ log.arsenic + log.dist + assoc + educ, data = training.wells)
#lda.baseline
#plot(lda.baseline)

lda.rmassoc=lda(switch ~ log.arsenic + log.dist + educ, data = training.wells)
#lda.rmassoc
#plot(lda.rmassoc)

# build predicted probabilities
lda.pred=predict(lda.baseline, testing.wells)
lda.class=lda.pred$class  # Predicted class assuming posterior prob > 0.5
```

```r
# build confusion matrix
t.lda <- table(lda.class,testing.wells$switch)
con.table.df.lda <- t(data.frame(t.lda[1,],t.lda[2,]))
rownames(con.table.df.lda) <- c(0,1)
library(knitr)
library(kableExtra)
library(dplyr)

kable(con.table.df.lda, "latex", caption = "Linear Discriminant Analysis Confusion Matrix", booktabs = 
kable_styling(latex_options = c("striped", "hold_position"))

# find error rate
#mean(lda.class==testing.wells$switch)
error.rate <- 1-(mean(lda.class==testing.wells$switch))
#error.rate
#sum(lda.pred$posterior[,2] >= 0.5) #Number of predictions that esrd = 1
#sum(lda.pred$posterior[,2] < 0.5) #Number of predictions that esrd = 0

# other way of building threshold mark effect (predicted 0's and 1's)
lda.testing.pred=rep(0,500)
lda.testing.pred[lda.pred$posterior[,2]>= 0.5]=1

#hist(lda.pred$posterior[,1])  #Histogram of posterior predictions

# other way to build confusion matrix
#table(lda.testing.pred,testing.wells$switch) #Predicted(column) vs. observed(row)
#mean(lda.testing.pred==training.wells$switch,na.rm=T)
library(ROCit)
# roc shenanigans
ROC_lda <- rocit(score=lda.pred$posterior[,2],class=testing.wells$switch)
plot(ROC_lda)
#summary(ROC_lda)
library(MASS)
# same model as above, to ask about variable selection
qda.baseline=qda(switch ~ log.arsenic + log.dist + assoc + educ, data = training.wells)
#qda.baseline

# build predicted probabilities
qda.pred=predict(qda.baseline, testing.wells)
qda.class=qda.pred$class  # Predicted class assuming posterior prob > 0.5

# build confusion matrix
t.qda <- table(qda.class,testing.wells$switch)
con.table.df.qda <- t(data.frame(t.qda[1,],t.qda[2,]))
rownames(con.table.df.qda) <- c(0,1)
library(knitr)
library(kableExtra)
library(dplyr)

kable(con.table.df.qda, "latex", caption = "Quadratic Discriminant Analysis Confusion Matrix", booktabs
kable_styling(latex_options = c("striped", "hold_position"))
```

```r
# find error rate
#mean(qda.class==testing.wells$switch)
error.rate <- 1-(mean(qda.class==testing.wells$switch))
#error.rate
#sum(lda.pred$posterior[,2] >= 0.5) #Number of predictions that esrd = 1
#sum(lda.pred$posterior[,2] < 0.5) #Number of predictions that esrd = 0

# other way of building threshold mark effect (predicted 0's and 1's)
#qda.testing.pred=rep(0,500)
#qda.testing.pred[qda.pred$posterior[,2]>= 0.5]=1

#hist(qda.pred$posterior[,1])  #Histogram of posterior predictions

# other way to build confusion matrix
#table(lda.testing.pred,testing.wells$switch) #Predicted(column) vs. observed(row)
#mean(lda.testing.pred==training.wells$switch,na.rm=T)
library(ROCit)
# roc shenanigans
ROC_qda <- rocit(score=qda.pred$posterior[,2],class=testing.wells$switch)
plot(ROC_qda)
#summary(ROC_qda)
library(class)
wells.X = cbind(wells$log.dist,wells$log.arsenic,wells$assoc,wells$educ)
standardized.X=scale(wells.X)
#var(wells$log.dist)
#var(wells$log.arsenic)
#var(wells$assoc)
#var(wells$educ)
#var(standardized.X[,1])
#var(standardized.X[,2])
#var(standardized.X[,3])
#var(standardized.X[,4])

training.wells <- wells[1:2520,]
testing.wells <- wells[2521:3020,]

train.X=standardized.X[1:2520,]
test.X=standardized.X[2521:3020,]
train.Y=wells[1:2520,1]
test.Y = wells[2521:3020,1]

set.seed(1)  #allows refitting same model since knn breaks ties randomly
knn.pred1=knn(train.X,test.X,train.Y,k=1)
#table(knn.pred1,test.Y)

# build confusion matrix
t.knn1 <- table(knn.pred1,test.Y)
con.table.df.knn1 <- t(data.frame(t.knn1[1,],t.knn1[2,]))
rownames(con.table.df.knn1) <- c(0,1)
library(knitr)
library(kableExtra)
library(dplyr)
```

```r
kable(con.table.df.knn1, "latex", caption = "KNN w K=1 Confusion Matrix", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))


#mean(knn.pred1==test.Y)
error.rate <- 1-(mean(knn.pred1==test.Y))
#error.rate

# K Nearest Neighbor with K = 5

knn.pred5=knn(train.X,test.X,train.Y,k=5)
#table(knn.pred5,test.Y)

# build confusion matrix
t.knn5 <- table(knn.pred5,test.Y)
con.table.df.knn5 <- t(data.frame(t.knn5[1,],t.knn5[2,]))
rownames(con.table.df.knn5) <- c(0,1)
library(knitr)
library(kableExtra)
library(dplyr)

kable(con.table.df.knn5, "latex", caption = "KNN w K=5 Confusion Matrix", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))

#mean(knn.pred5==test.Y)
error.rate <- 1-(mean(knn.pred5==test.Y))
#error.rate

library(knitr)
library(kableExtra)
library(dplyr)

#build data frame of summary stats
summ.stats <- data.frame(c(0.8376, 0.3045, "45%", "0.6498", 3),
                         c(0.8376, 0.3007, "44.8%", "0.6505", 2),
                         c(0.8205, 0.3571, "42.6%", "0.6696", 1),
                         c(0.6196, 0.4511, "47%", NA, 5),
                         c(0.7094, 0.3947, "45.8%", NA, 4))


summ.stats <- t(summ.stats)
colnames(summ.stats) <- c("Sensitivity", "Specificity", "Error Rate", "AUC", "My Ranking")
rownames(summ.stats) <- c("Logistic Regression",
                          "Linear Discriminant",
                          "Quadratic Discriminant",
                          "KNN w/ k=1",
                          "KNN w/ k=5")

#build table
kable(summ.stats, "latex", caption = "Model Performance Comparison", booktabs = T) %>%
kable_styling(latex_options = c("striped", "hold_position"))
```