PDA Assignment #4

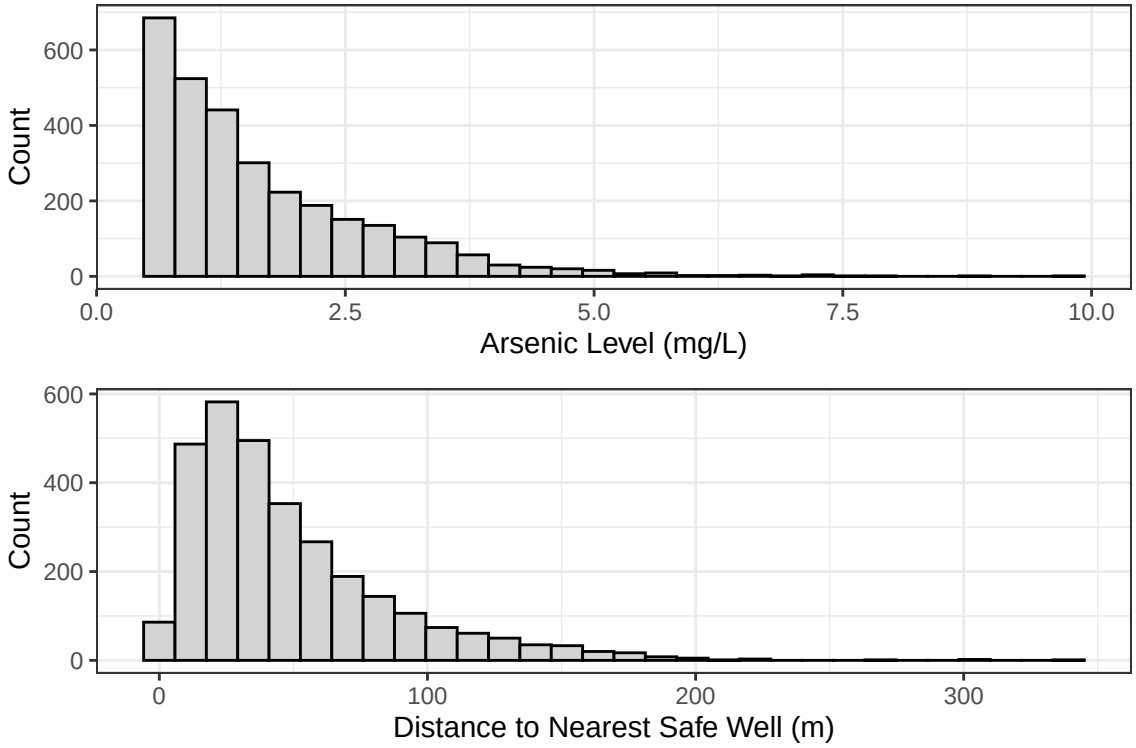
Anthony Sisti

11/01/2019

Problem 1

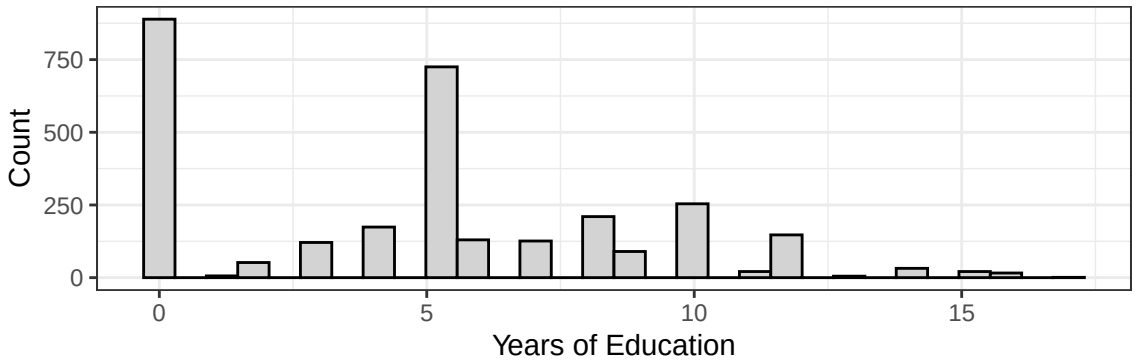
Exploratory Data Analysis

We begin by conducting some exploratory data analysis on the 'Wells' data set. We first plot the two continuous variables, 'arsenic' and 'dist'.

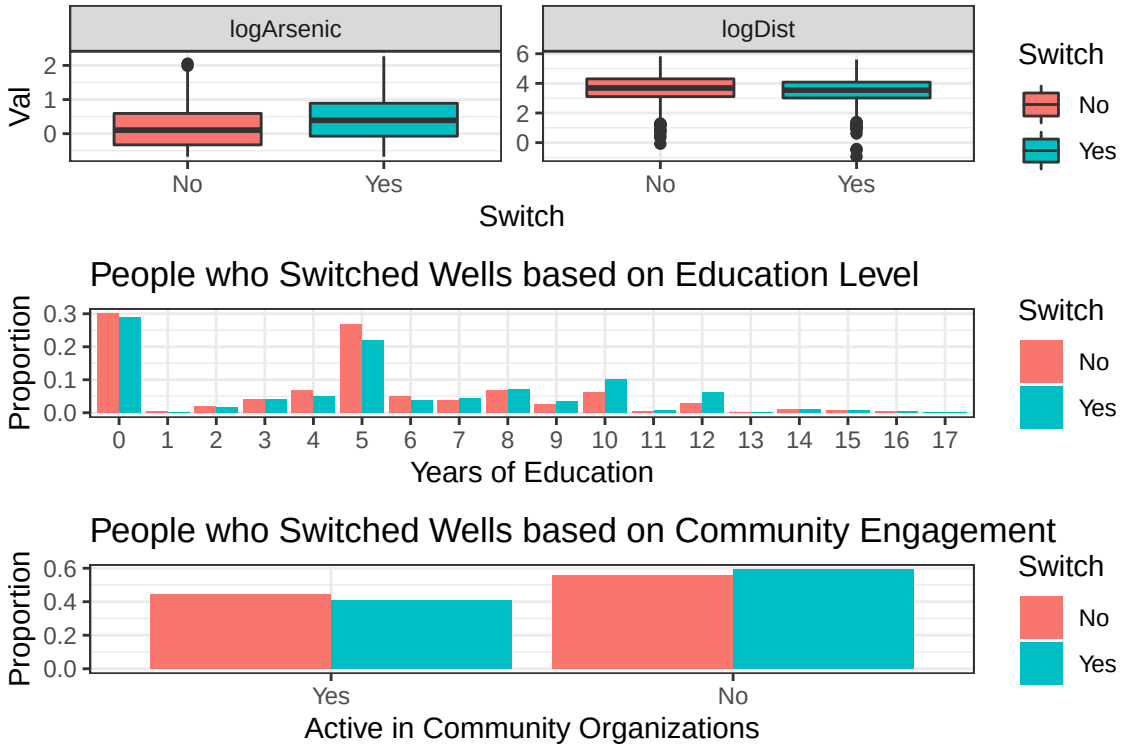


These variables appear to follow an exponential decay. In order to normalize the variables, ensuring the best possible results for regression and prediction, we take the log of both.

Now we plot the variable describing education.



We see the most educated member in the families most frequently had no education or 5 years of education. This is clearly not a normally distributed variable. Next, we visualize the relationship between the variables and whether or not an individual switched wells. Box plots for our continuous predictors, and side by side bar plots for the categorical predictors are displayed below.



The box plots show that in general, higher arsenic values correspond to switching more often, while being a farther distance from the nearest safe well makes a person generally less likely to switch. The first side by side bar plots show the proportion of families who switched wells or didn't that fell into each educational category. It appears that a slightly larger proportion of those who did not switch are clustered at lower education levels and a slightly larger proportion of those who switched are clustered at higher education levels. This exemplified in the difference in proportion in the 5 year and 10 year education category.

Logistic Regression

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.

b. Compute and graph the predicted probabilities stratifying by the predictors. 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.

We combine part a) and b) to pick the best model. We begin constructing a logistic regression to predict whether a family switched wells with the following factors.

1. Education - Highest year of education in the family
2. Association - 1 Family involved in community organization, 0 Not
3. logDist - Natural log Distance (m) to nearest safe well
4. logArsenic - Natural log Arsenic (mg/L) found in the well the family was using

We first fit a full logistic regression model containing all four variables. The model is fit on the training set of the first 2520 observations from the 'Wells' data set. The summary of this model is shown below.

Table 1: Full Fit all Predictors

	Dependent variable:
	Switch
Education	0.041^{***}
	(0.011)
logDist	-0.360^{***}
	(0.053)
logArsenic	0.828^{***}
	(0.075)
Association	-0.169^{**}
	(0.085)
Constant	1.318^{***}
	(0.201)
Observations	2,520
Log Likelihood	−1,616.348
Akaike Inf. Crit.	3,242.697
Note:	*p<0.1; **p<0.05; ***p<0.01

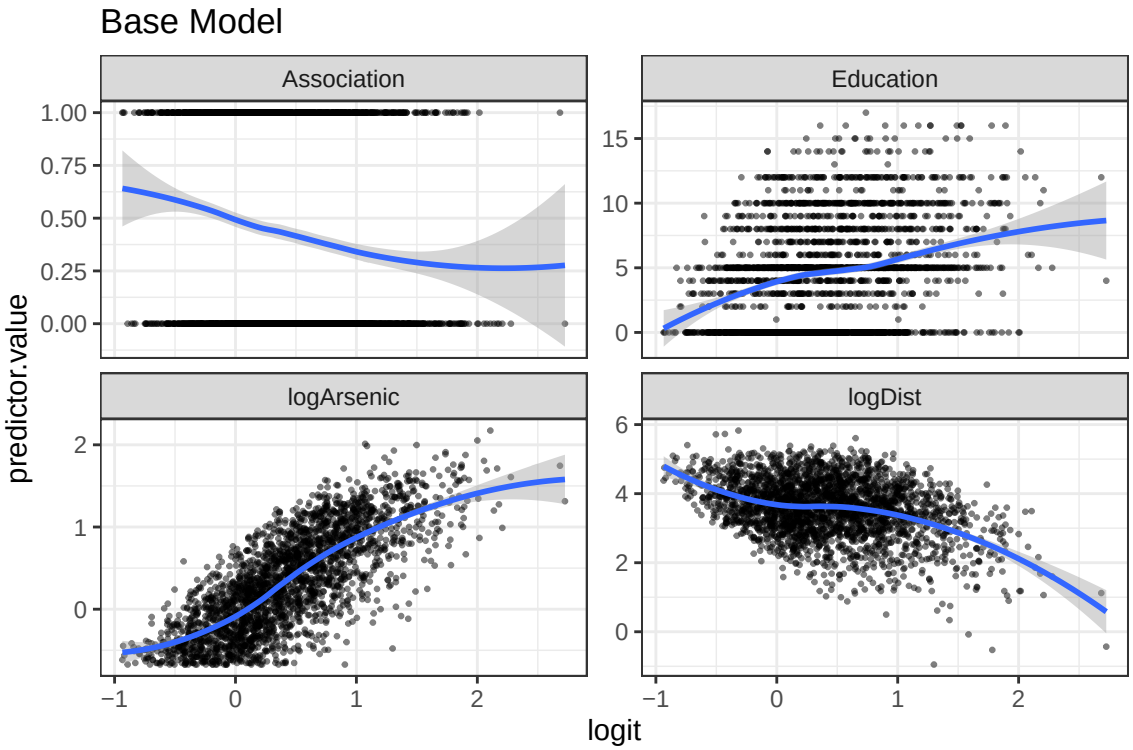
All four predictor coefficients are significantly different from zero at the 5% level. The model summary tells us that higher Arsenic levels and more years education increase the the odds that a family will switch wells,

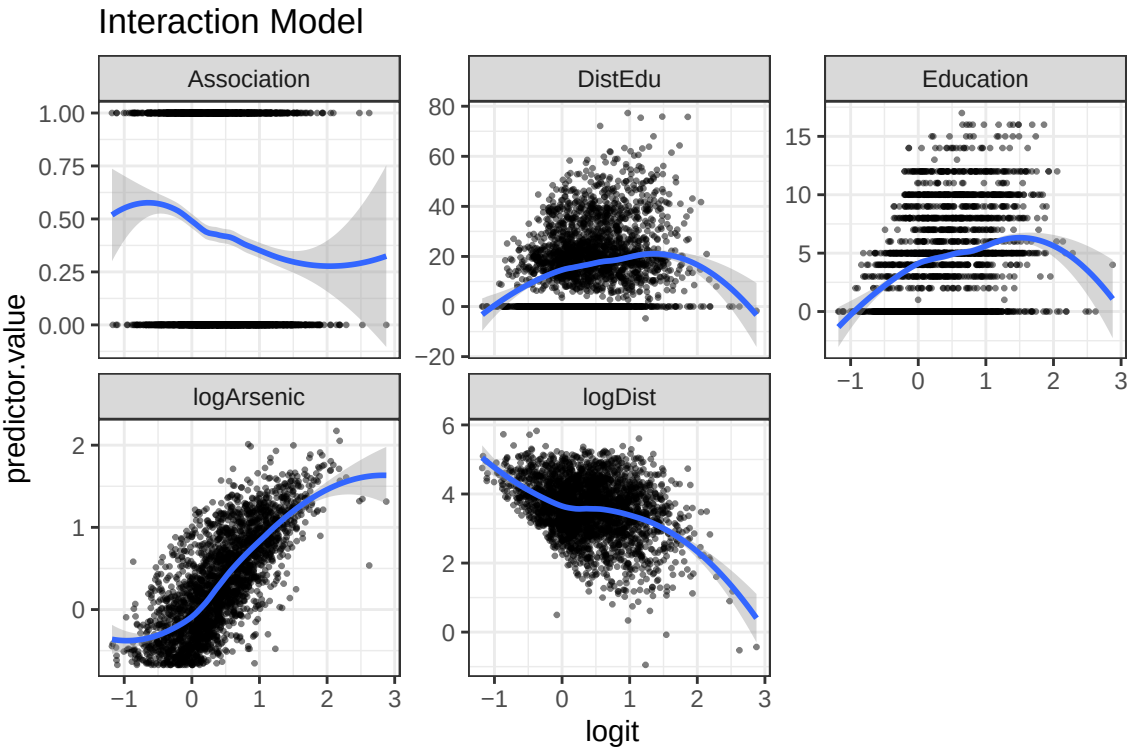
while greater distance and stronger community ties tend to decrease the odds of switching. This follows the trends we observed in the EDA. Since all the variables are significant predictors of whether a family will switch, we continue to build the model with all of them included. We next consider adding a two way interaction to the model. To evaluate the usefulness of each interaction, we construct six new models, each building off the original fit and containing one of the six possible two way interaction terms. Below, we compare the BIC and AIC values of these models.

Table 2: Comparison of Models: Single Two-way Interaction

	df	BIC	AIC
Basic_fit	5	3271.857	3242.697
ArsDist	6	3276.248	3241.256
ArsAssoc	6	3278.960	3243.968
ArsEdu	6	3274.238	3239.246
DistAssoc	6	3277.424	3242.432
DistEdu	6	3268.204	3233.212
EduAssoc	6	3279.285	3244.293

We see that the model containing the interaction between 'logDistance' and 'Education' is the only model that improves on both BIC and AIC from the basic fit, albeit slightly. Since adding an interaction only marginally reduces AIC and BIC values, we consider both the base and interaction model in our regression diagnostics, and choose the better of the two from based on the analysis. Plots of the logit value by predictor for each model are presented below.





Based on the plots above, the interaction model does not meet the linearity assumption. Each predictor deviates farther from linearity than in the base model, and the interaction term has a particularly non-linear relationship with log odds. The only variable in the base model that could need transformation is 'logArsenic' as it shows slight non-linearity at its tail values, but since there are very few values exhibiting deviations from linearity, it approximately linear. We also consider the collegiality and standardized residuals of each model below.

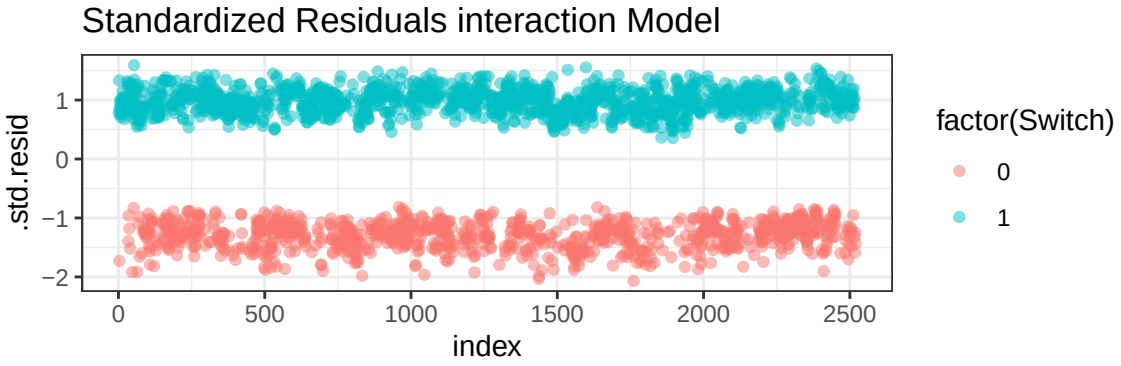


Table 3: Variance Inflation Factor : Base Model

	VIF
Education	1.003
logDist	1.098
logArsenic	1.100
Association	1.001

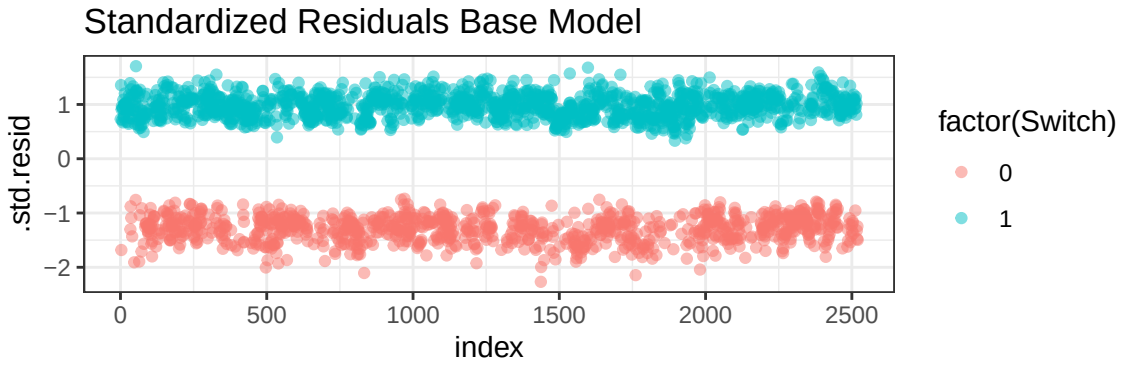
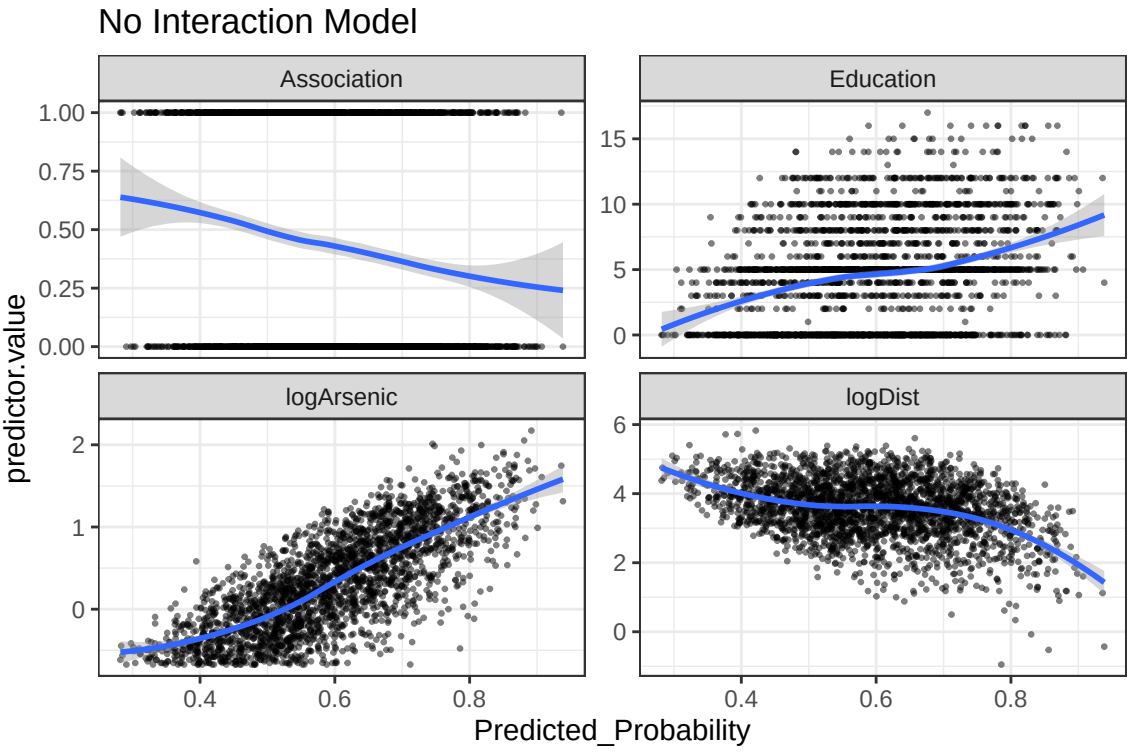


Table 4: Variance Inflation Factor : Interaction Model

	VIF
logArsenic	1.105
logDist	2.600
Association	1.003
Education	19.518
logDist:Education	20.438

While it doesn't appear that there are any severe outliers in either population, or extremely influential values with standardized residual greater than three, the VIFs found in the model with the interaction are very large. Namely, the VIF for the interaction and 'Education' variable are around 20. This is indicative of large confounding effects between predictors. Adding this realization to what we visualized in the probability plots previously, we determine it is best to remove the interaction and remain with the base model for prediction. We plot the probabilities based on the value of the predictors below.



This plot closely resembles the logit plot for the no interaction model. The a lack of association with the community tends to lead to higher probability that the family will switch wells, as does more years of education and higher arsenic levels. As distance from the nearest safe well increases, the probability of the family making a switch tends to decrease.

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.)

Below we create the confusion matrix using a cutoff of p = 0.5.

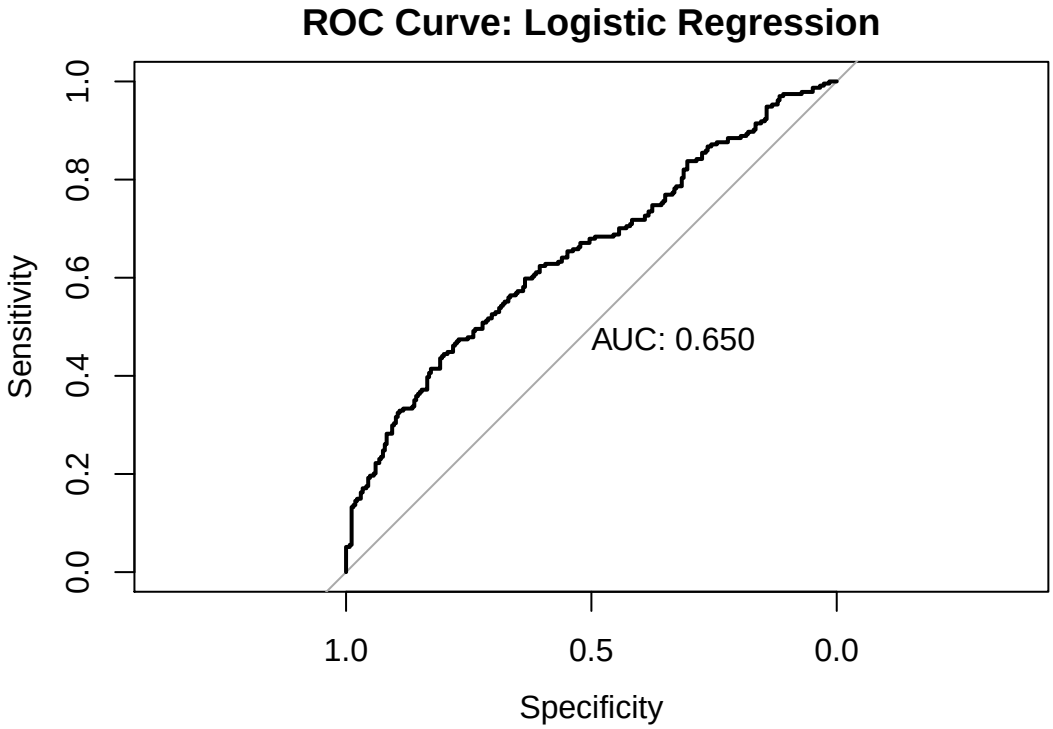
Table 5: Confusion Matrix

	Predicted 0	Predicted 1	Total
Actual 0	81	185	266
Actual 1	38	196	234
Total	119	381	500

The error rate for our model is 45%. Taking 1, the group that switched wells, to be our positive case, the sensitivity is 84% and the specificity is 30%. Based on these rates, our model is not very accurate, only slightly better than random guessing. It is also better at predicting a value when the person switched rather than when the person did not switch. If a switch occurred, there is a good chance that you will identify it as a switch, but if a switch did not occur, there is a good chance that that will be classified as a switch as well.

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

Below we construct an ROC plot, and find the area under the curve.



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?)

Our current threshold of 0.5 puts us high on the ROC curve, with sensitivity 0.84 and specificity 0.29. To balance sensitivity and specificity, we need to increase the threshold to make it more difficult to classify a predictor as a "switch". This way we would reduce false positives while increasing true negatives. This would bring sensitivity and specificity closer together. The best we could do to balance it, while maintaining some advantage over random guessing, would be to increase alpha to the point that sensitivity is slightly above 60% and specificity is around 55%. Overall, this would still not be a very effective model.

LDA

a. Construct a good LDA 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.

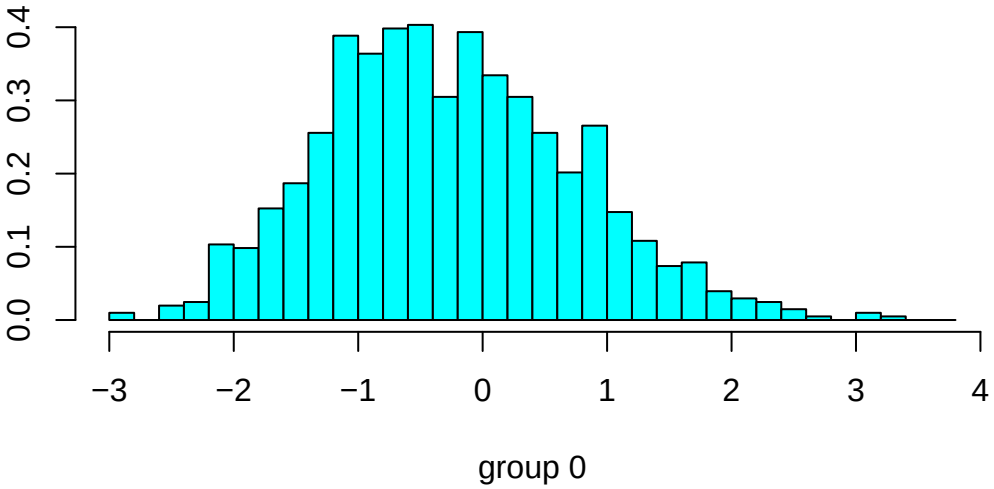
We now conduct a similar analysis using LDA. LDA assumes multivariate normal predictors with the same variance between groups and categorical response. We decide to drop the 'Education' and 'Association' predictors because they clearly deviate from this assumption. We fit our model with standardized box-cox transforms of 'dist' and 'arsenic' to ensure they are as close as possible to normality. For the arsenic level variable, $\lambda = -0.21$ and for the distance variable, $\lambda = 0.21$. The linear discriminant coefficients for each variable can be found below.

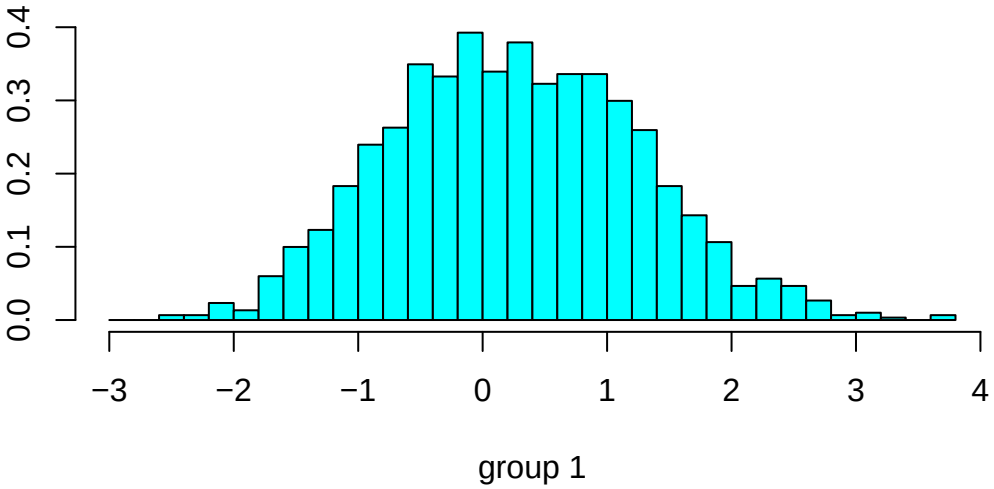
Table 6: LDA coefficients for transformed arsenic and distance variables

	LD1
arsenic	0.9816664
dist	-0.6445300

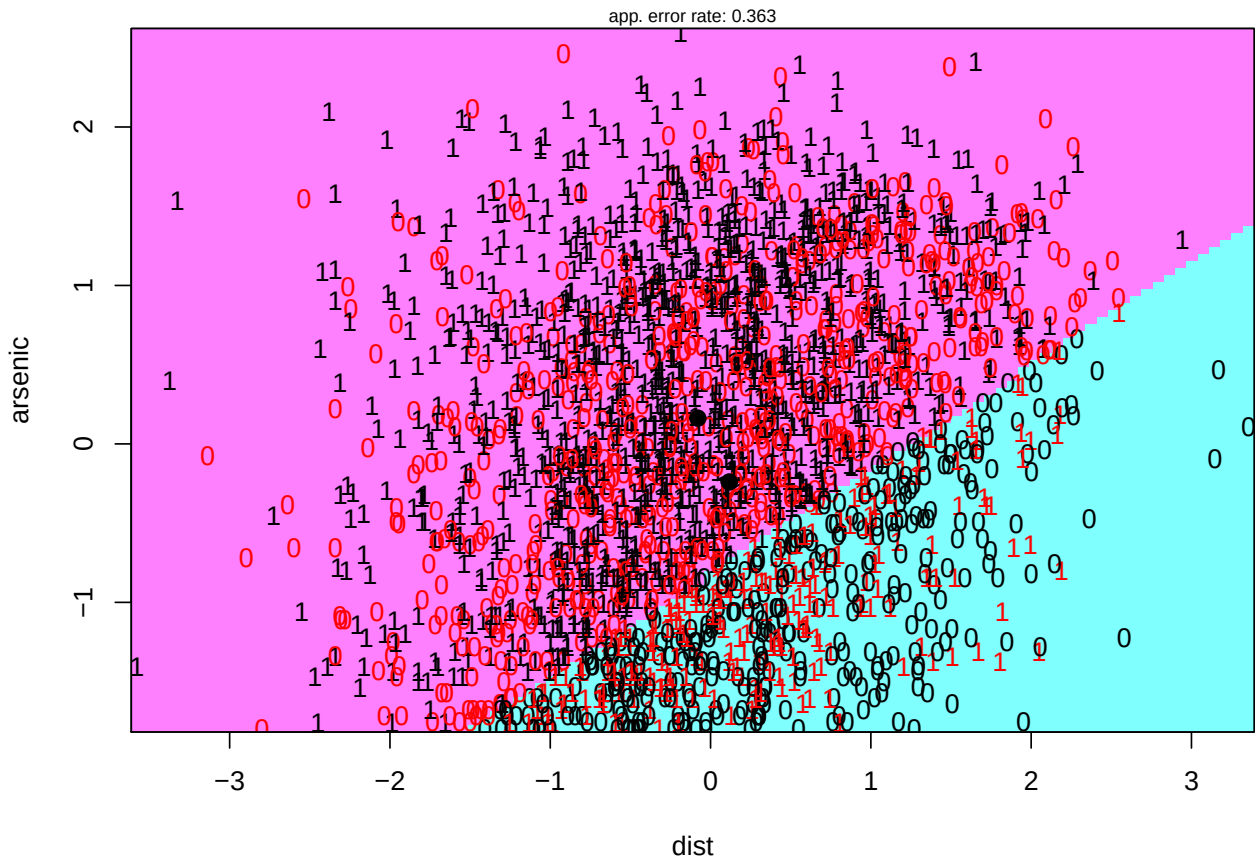
b. Compute and graph the predicted probabilities stratifying by the predictors. You could do this using graphs such as in the papers we discussed in class or by using contour plots which would allow you to graph two continuous predictors on the same plot. You can array different lines and plots to try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

To address this question, we plot the distributions of our groups with respect to the linear discriminant.





The group of people who switched wells, group 1, has a slightly higher mean value when compared to the group of people that did not switch wells, group 0. Although there is a slight difference, there is a large overlap between the two distributions, likely making it difficult to distinguish the two classes. We visualize the linear separation between the classes in our training data below.



Clearly, there is too much overlap to distinctly separate classes, but on the training data, the model better than random guessing with an error rate of 36%.

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.)

Below, we display the confusion matrix for our LDA model predicting on the test data.

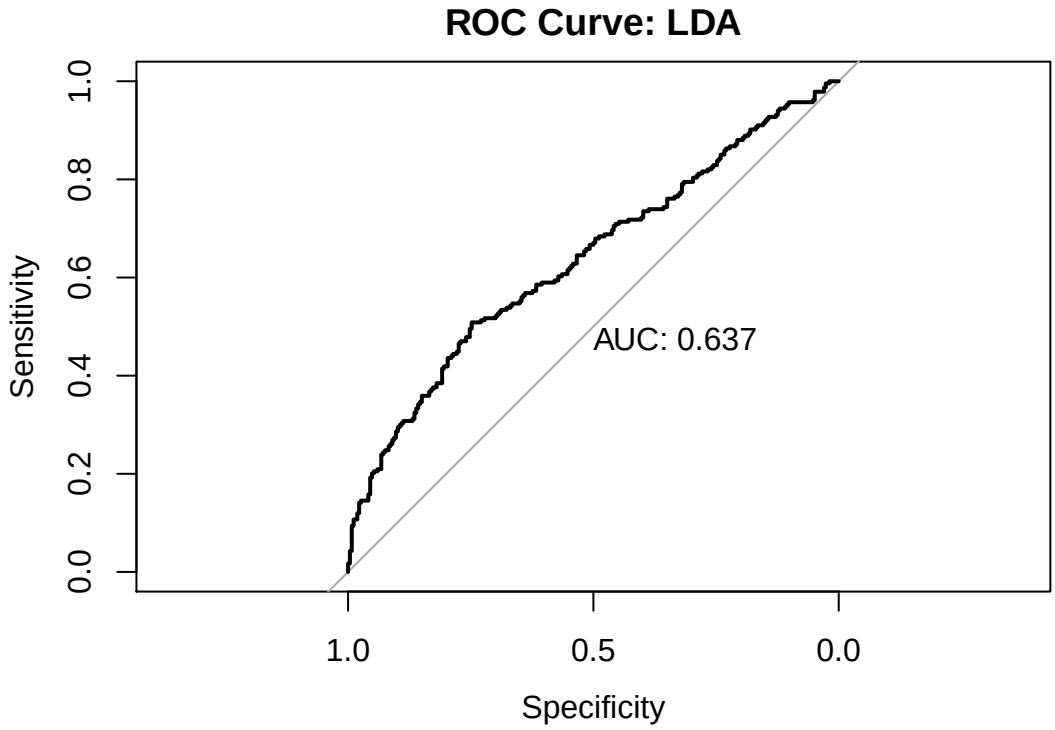
Table 7: Confusion Matrix : LDA

	Actual 0	Actual 1
0	75	44
1	191	190

The error rate of the model is 47%. Taking 1, the group who switched wells, as the positive class, our sensitivity is 81% and our specificity is 28%. This indicates that this model is, again, far better at predicting positive cases than negative cases. An actual positive will likely be predicted as such, but an actual negative is more likely to be predicted as a positive.

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

Below we construct an ROC plot and print the area under the curve.



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?)

Like the logistic regression, the current 0.5 threshold places us high on the curve indicating high sensitivity and low specificity. In order to balance the sensitivity and specificity we would have to increase the threshold. The best we could likely do is to have both sensitivity and specificity around 55%. The small area under the curve, 0.637, indicates that the model is not very accurate overall.

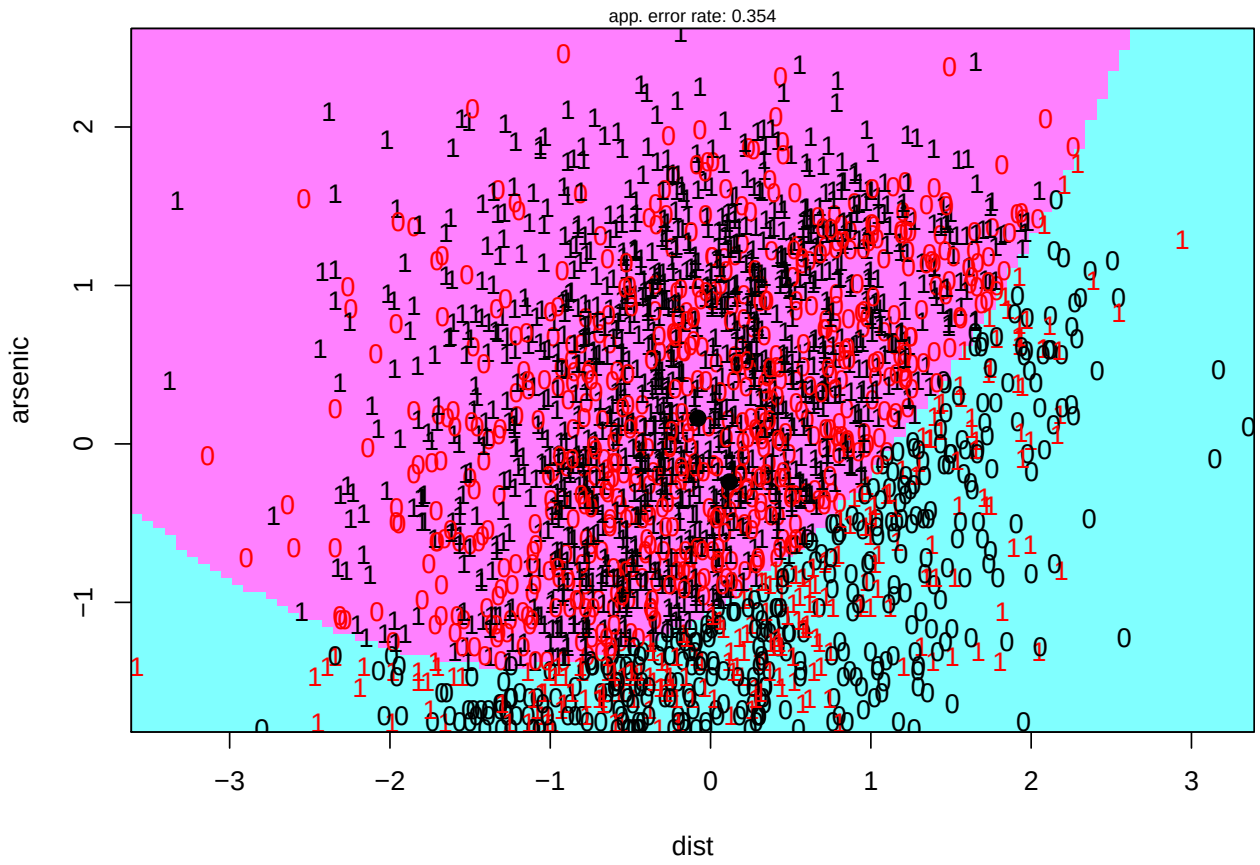
QDA

a. Construct a good QDA 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.

We now conduct the analysis using QDA. Like LDA, QDA assumes multivariate normal predictors, but not necessarily with the same covariance structure across the response. We again decide to drop the 'Education' and 'Association' predictors because they clearly deviate from this assumption. We fit our model with the standardized box-cox transforms of 'dist' and 'arsenic' like in LDA.

b. Compute and graph the predicted probabilities stratifying by the predictors. 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.

We visualize the quadratic separation between the classes in our training data below.



The variance and overlap within the classes makes it difficult to distinctly separate them, but on the training data, the model better than random guessing with an error rate of 35.4%.

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.)

Below, we display the confusion matrix for our LDA model predicting on the test data.

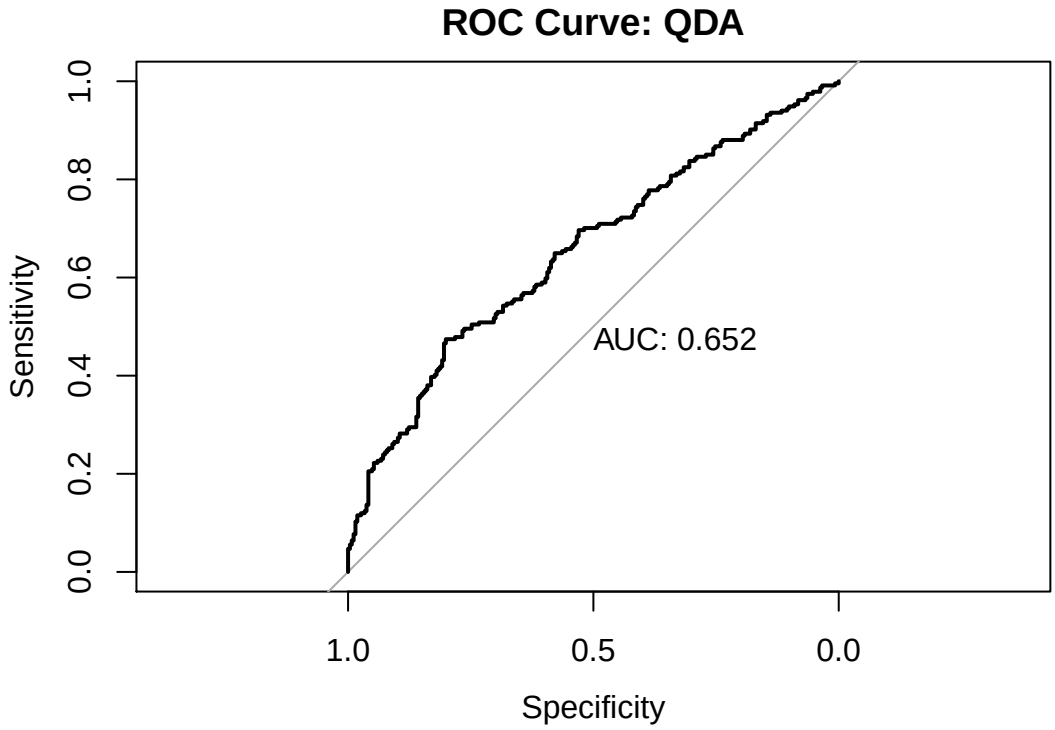
Table 8: Confusion Matrix : LDA

	Actual 0	Actual 1
0	84	41
1	182	193

The error rate of the model is 45%, slightly better than LDA. Taking 1, the group who switched wells, as the positive class, our sensitivity is 82.5% and our specificity is 31.5%. This indicates that this model is, yet again, far better at predicting positive cases than negative cases. This model is slightly better at predicting true negatives as negatives, but it is still well below 50%.

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

Below we construct an ROC plot and print the area under the curve.



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?)

Like the logistic regression and LDA, the current 0.5 threshold places us high on the curve indicating high sensitivity and low specificity. In order to balance the sensitivity and specificity we would have to increase the threshold. The best we could likely do is to have both sensitivity around 60% and specificity around 55%, based on the curve. The area under the curve, 0.652, is similar to logistic regression and slightly higher to LDA. Still, this indicates that the model is not very accurate regardless of the threshold we choose.

KNN: K=1 and K=5

Based on the earlier EDA, we believe it would be best to use log transformations of our 'dist' and 'arsenic' variables for KNN prediction. Furthermore, to prepare the data we standardize 'logDist', 'logArsenic' and 'Education' , while leaving 'Association' (0,1 indicator) as is. Since there are no probabilities from the KNN, we produce the KNN output with K=1 and print the confusion matrix.

Table 9: Confusion Matrix: KNN, K=1

	Actual 0	Actual 1
0	116	75
1	150	159

The confusion matrix for $K = 1$ indicates that we have an error rate of 45% on our test data. Taking 1 to be our positive case, we have a sensitivity of 68% and a specificity of 44%. This model is slightly more balanced than our previous three, as sensitivity and specificity are closer. There also a similar overall error rate to the

other models. Now we bud the same model where $K = 5$.

Table 10: Confusion Matrix: KNN, K=5

	Actual 0	Actual 1
0	107	57
1	159	177

The confusion matrix for $K = 5$ indicates that we have an error rate of 43% on our test data. Taking 1 to be our positive case, we have a sensitivity of 76% and a specificity of 40%. This model is slightly less balanced than $K = 1$ but still more balanced than our previous three models. There also is a smaller overall error rate than the other three types of models.

Summary

Below we summarize some of our results for each model.

Table 11: Comparison of Models

Model	ErrorRate	Sensitivity	Specificity	AUC
Logistic Regression	0.45	0.840	0.300	0.65
LDA	0.47	0.810	0.280	0.637
QDA	0.45	0.825	0.315	0.652
KNN1	0.45	0.680	0.440	
KNN5	0.43	0.760	0.400	

The model with the lowest error rate was KNN when $K = 5$. It was only slightly better than each of the other models. If your main concern was determining true positives, it would be best to use logistic regression, as it had the highest sensitivity on the test data. Similarly, if your concern was true negatives, you would be most inclined to use KNN with $K = 1$, since it had a far higher specificity than any of the non KNN methods. The model with the highest AUC value was QDA, but is worth noting that KNN is not comparable by this metric. Overall, depending on the context of your study, different models could have provided different benefits. In general, however, it appears we would need different methods to achieve quality predictions for this data set.

Problem 2

Do exercise 2.9 in Hastie and Tibshirani.

Let $\hat{\beta}$ be the training set least squares estimate, and define $\tilde{\beta}$ to be the least squares estimate for the new test set. By the definition of the least squares estimator we have,

$$\frac{1}{M}\sum_{i=1}^{M}(\tilde{y_i} - \tilde{\beta}^T\tilde{x_i})^2 \leq \frac{1}{M}\sum_{i=1}^{M}(\tilde{y_i} - \hat{\beta}^T\tilde{x_i})^2$$

and so

$$E\left[\frac{1}{M}\sum_{i=1}^{M}(\tilde{y_i}-\tilde{\beta}^T\tilde{x_i})^2\right] \leq E\left[\frac{1}{M}\sum_{i=1}^{M}(\tilde{y_i}-\hat{\beta}^T\tilde{x_i})^2\right]$$
$$E\left[R_{te}(\tilde{\beta})\right] \leq E\left[R_{te}(\hat{\beta})\right]$$

This is not particular to $\hat{\beta}$, but would follow for any other coefficient estimate. We are left to show that $E[R_{tr}(\hat{\beta})] = E[R_{te}(\tilde{\beta})]$. We know that within the training sample,

$$E\left(\left(y_i-\hat{\beta^T}x_i\right)^2\right)=\sigma^2$$

where σ^2 is the underlying error variance. This is valid since were are taking the expectation of y_i and x_i, which are random pulls from a population that we assume a model $y = \beta x + \epsilon$. So we expect this relationship to hold for any given (x, y) pairing. Since $\hat{\beta}$ is the best unbiased estimator for β on this set of data, the expected squared error is simply the error variance. Within the training data we also have

$$E\left(\left(\tilde{y_i}-\tilde{\beta}^T\tilde{x_i}\right)^2\right)=\sigma^2$$

since, on the test data, $\tilde{\beta}$ should behave the same as $\hat{\beta}$ would on the test data, minimizing squared error, and when taking expectation over the entire sample, this squared error should average out to the error variance. Thus,

$$E\left[R_{te}(\tilde{\beta})\right]=\sigma^2=E[R_{tr}(\hat{\beta})]$$

proving that $E\left[R_{te}(\hat{\beta})\right] \leq E\left[R_{te}(\hat{\beta})\right]$.

Problem 3

Assume that y = f(X) + e is the true model. You have estimated f by a function g. Show that the square of the expected error E(Y-g(X)] equals the sum of the variance of g(X), the square of the bias of g(X), and the variance of the irreducible error from e.

We have

$$\begin{aligned}
E\left[(Y-g(X))^2\right] &= Var[Y-g(X)]+E[(Y-g(X))]^2 \\
&= Var\left[(f(X)-g(X))+\epsilon\right]+E[(f(X)-g(X))+\epsilon]^2 \\
&= Var[(f(X)-g(X))]+Var[\epsilon]+E[(f(X)-g(X))]^2-2E[\epsilon]E[(f(X)-g(X))]+E[\epsilon]^2 \\
&= Var[(f(X)-g(X))]+Var[\epsilon]+E[(f(X)-g(X))]^2 \\
&= Var[g(X)]+Var[\epsilon]+E[(f(X)-g(X))]^2
\end{aligned}$$

Where the last line comes from the fact that $f(X)$ is not random, rather, it is the parameter we are estimating with $g(X)$. The final equality is the variance of our estimator, the variance of our irreducible error and the squared bias of our estimator.

Problem 4

Given inputs x and outputs y and a model f(x) fit by least squares, show that if there are observations with identical values of x, then the fit can be obtained from weighted least squares with specific weights.problem.

The the least squares problem involves minimizing

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - x_i^T \beta)^2$$

Assume $x_N = x_{N-1}$ without loss of generality. We can change the problem into weighted least squares if we define

$$RSS(\beta) = \sum_{i=1}^{N-1} w_i (y_i - x_i^T \beta)^2$$

where all $w_i = 1$, except $w_{N-1} = 2$. Since $x_N = x_{N-1}$, our assumptions tell us we expect $y_N = y_{N-1}$, so the problems are equivalent, given a random sample with this property.

Code Appendix

```
knitr::opts_chunk$set(echo = F)
knitr::opts_chunk$set(message = F)
knitr::opts_chunk$set(warning = F)
knitr::opts_chunk$set(fig.height = 4)
knitr::opts_chunk$set(fig.width = 6)
knitr::opts_chunk$set(fig.align="center")
Wells = read.table("/Users/Anthony/Documents/Brown 1/Practical Data Analysis/HW4/wells.txt")
library(tidyverse)

#Plot arsenic, distance and education variable

pars = ggplot(Wells, aes(x = arsenic))+ geom_histogram(color = "black", fill = "lightgrey") +theme_bw()
  ylab("Count") + xlab("Arsenic Level (mg/L)")

pdist = ggplot(Wells, aes(x = dist))+ geom_histogram(color = "black", fill = "lightgrey") +theme_bw() +
  ylab("Count") + xlab("Distance to Nearest Safe Well (m)")

gridExtra::grid.arrange(pars, pdist)


#Rename columns and take log of approprate columns from previous plots

NewWells = Wells %>%
  dplyr::rename(logArsenic = arsenic , logDist = dist, Education = educ, Association = assoc, Switch = s

NewWells[, c(2,3)] = log(NewWells[, c(2,3)])

FacetWells = NewWells %>%
  gather("Var", "Val", 2:3)

NewWells$Switch = factor(NewWells$Switch, levels = c(0,1), labels = c("No", "Yes"))
FacetWells$Switch = factor(FacetWells$Switch, levels = c(0,1), labels = c("No", "Yes"))


pars = ggplot(Wells, aes(x = educ))+ geom_histogram(color = "black", fill = "lightgrey") +theme_bw() +
  ylab("Count") + xlab("Years of Education")

pars


# EDA seeing how the response compares to our predictors
```

```r
# Boxplots for binary predictors and barplots for numerical

p1 = ggplot(FacetWells, mapping = aes(x = Switch, y = Val, fill = Switch )) + geom_boxplot()+
  facet_wrap(~Var, scales = 'free') + theme_bw()


p2= ggplot(NewWells, aes(fill=Switch, x=factor(Education), group = Switch )) +
    geom_bar(aes(y = (..prop..)), position = "dodge") +xlab("Years of Education")+
  ylab("Proportion")+ggtitle("People who Switched Wells based on Education Level") +theme_bw()

p3= ggplot(NewWells, aes(fill=Switch, x=factor(Association, levels = c(1,0), labels = c("Yes", "No")),
                         group = Switch )) + geom_bar(aes(y = (..prop..)), position = "dodge") +
  xlab("Active in Community Organizations")+
  ylab("Proportion")+ggtitle("People who Switched Wells based on Community Engagement")+ theme_bw()

gridExtra::grid.arrange(p1,p2,p3)



#Create Test and Training set

NewWells$Switch = ifelse(NewWells$Switch == "Yes", 1, 0)
TrainWells = NewWells[1:2520,]
TestWells = NewWells[2521:3020,]

# Fit Full Model

Basic_fit = glm(Switch~ Education + logDist + logArsenic + Association , data = TrainWells, family = bi


#Show basic logistic regression results
library(stargazer)
stargazer(Basic_fit, header = F, title = "Full Fit all Predictors")

##Compare models with each possible interaction

library(kableExtra)
ArsDist = glm(Switch ~ logArsenic + logDist + Association + Education + logArsenic*logDist  ,
              data = TrainWells, family = binomial())

ArsAssoc = glm(Switch ~ logArsenic + logDist + Association + Education + logArsenic*Association  ,
               data = TrainWells, family = binomial())
```

```r
ArsEdu = glm(Switch ~ logArsenic + logDist + Association + Education + logArsenic*Education  ,
             data = TrainWells, family = binomial())

DistAssoc = glm(Switch ~ logArsenic + logDist + Association + Education + logDist*Association  ,
                data = TrainWells, family = binomial())

DistEdu = glm(Switch ~ logArsenic + logDist + Association + Education + logDist*Education  ,
              data = TrainWells, family = binomial())

EduAssoc = glm(Switch ~ logArsenic + logDist + Association + Education + Education*Association  ,
               data = TrainWells, family = binomial())

kable_styling(kable(cbind(BIC(Basic_fit, ArsDist, ArsAssoc, ArsEdu, DistAssoc, DistEdu, EduAssoc),
      AIC(Basic_fit, ArsDist, ArsAssoc, ArsEdu, DistAssoc, DistEdu, EduAssoc))[,-3],
      caption = "Comparison of Models: Single Two-way Interaction"),latex_options = "HOLD_position")



library(dplyr)

# logistic regression diagnostics

TrainWells$DistEdu = TrainWells$logDist*TrainWells$Education

# Get predicted probs for the no interation and interactio model

Noint_probs <- predict(Basic_fit, type = "response")
int_probs <- predict(DistEdu, type = "response")


#Create a data frame of predictors for both models

logistic_predictor_noint_pre <- TrainWells %>% dplyr::select("logArsenic", "logDist", "Association", "E
predictors_noint <- colnames(logistic_predictor_noint_pre)

logistic_predictor_int_pre <- TrainWells %>% dplyr::select("logArsenic", "logDist", "Association",
                                                           "Education", "DistEdu")
predictors_int <- colnames(logistic_predictor_int_pre)


#Add log odds Column

logistic_predictor_noint <- logistic_predictor_noint_pre %>%
```

```
 mutate(logit = log(Noint_probs/(1-Noint_probs))) %>%
 gather(key = "predictors_noint", value = "predictor.value", -logit)

logistic_predictor_int <- logistic_predictor_int_pre %>%
 mutate(logit = log(int_probs/(1-int_probs))) %>%
 gather(key = "predictors_int", value = "predictor.value", -logit)


#Plot Log odds by predictor

p_noint = ggplot(logistic_predictor_noint, aes(logit, predictor.value))+
geom_point(size = 0.5, alpha = 0.5) + geom_smooth(method ="loess") +
  theme_bw() + facet_wrap(~predictors_noint, scales = "free_y") + ggtitle("Base Model")

p_int = ggplot(logistic_predictor_int, aes(logit, predictor.value))+
geom_point(size = 0.5, alpha = 0.5) + geom_smooth(method ="loess") +
  theme_bw() + facet_wrap(~predictors_int, scales = "free_y") + ggtitle("Interaction Model")
p_noint
p_int
library(broom)

# Show standardized residuals and VIF factors

Basic_data =  broom::augment(Basic_fit) %>%
  dplyr::mutate(index = 1:n())

ggplot(Basic_data, aes(index, .std.resid)) +
  geom_point(aes(color = factor(Switch)), alpha = .5) +
  theme_bw() + ggtitle("Standardized Residuals interaction Model")


kable_styling(kable(round(car::vif(Basic_fit),3), caption = "Variance Inflation Factor : Base Model",
                    col.names = c("VIF")), latex_options = "HOLD_position")
# Show standardized residuals and VIF factors

Interaction_data =  broom::augment(DistEdu) %>%
  dplyr::mutate(index = 1:n())

ggplot(Interaction_data, aes(index, .std.resid)) +
  geom_point(aes(color = factor(Switch)), alpha = .5) +
  theme_bw() + ggtitle("Standardized Residuals Base Model")
```

```
kable_styling(kable(round(car::vif(DistEdu),3), caption = "Variance Inflation Factor : Interaction Model
                    col.names = c("VIF")), latex_options = "HOLD_position")


#graph probabilities as a funcition of the predictors

Predicted_Probability = Noint_probs
logistic_predictor_noint_pre = cbind(logistic_predictor_noint_pre, Predicted_Probability)


logistic_predictor_noint_pre <- logistic_predictor_noint_pre %>%
 gather(key = "predictors_noint", value = "predictor.value", -Predicted_Probability)
probsgraph = ggplot(logistic_predictor_noint_pre, aes(Predicted_Probability, predictor.value))+
geom_point(size = 0.5, alpha = 0.5) + geom_smooth(method ="loess") +
  theme_bw() + facet_wrap(~predictors_noint, scales = "free_y") + ggtitle("No Interaction Model")

probsgraph

library(regclass)

#Create Confusion Matrix

PredProb =predict.glm(Basic_fit, type=c("response"), newdata = TestWells )
TestWells$prob=PredProb

conf_mat = data.frame(confusion_matrix(Basic_fit, DATA = TestWells))

kable_styling(kable(conf_mat, caption ="Confusion Matrix", col.names = c("Predicted 0", "Predicted 1","
              latex_options = "HOLD_position")



#ROC Curve

library(pROC)
RocCurve <- roc(Switch ~ prob, data = TestWells)
plot(RocCurve, main = "ROC Curve: Logistic Regression", print.auc = T)

### LDA

library(MASS)
library(bestNormalize)
```

```r
LDATrainWells = Wells[1:2520,]
LDATestWells = Wells[2521:3020,]



toScalearsenic = LDATrainWells$arsenic
toScaledist = LDATrainWells$dist

test = scale((toScalearsenic^(-0.211) - 1) / -0.211)

BCarsenic = bestNormalize::boxcox(toScalearsenic)
BCdist = bestNormalize::boxcox(toScaledist)

LDATrainWells$arsenic = BCarsenic$x.t
LDATrainWells$dist = BCdist$x.t

#BoxCox transform of data to ensure normality

LDATestWells$arsenic = as.vector(scale(((LDATestWells$arsenic^(-0.21) - 1)/-0.21)))
LDATestWells$dist = as.vector(scale(((LDATestWells$dist^(0.21) - 1)/0.21)))



#Run LDA model

LDAWellsModel = lda(switch~ arsenic + dist, data = LDATrainWells)

kable_styling(kable(coef(LDAWellsModel),
                    caption = "LDA coefficients for transformed arsenic and distance variables"),
              latex_options = "HOLD_position")



plot(LDAWellsModel)


#Plot linear split
library(klaR)
partimat(factor(switch)~arsenic+dist, data=LDATrainWells, method="lda")


# Create Confusion Matrix
LDApredictions = LDAWellsModel %>% predict(LDATestWells)

LDAConfusion = caret::confusionMatrix(LDApredictions$class, factor(LDATestWells$switch), positive = "1"
```

```
kable_styling(kable(LDAConfusion$table, caption = "Confusion Matrix : LDA",
                    col.names = c("Actual 0", "Actual 1")),latex_options = "HOLD_position")


LDATestWells$prob = LDApredictions$posterior[,2]
LDARocCurve <- roc(switch ~ prob, data = LDATestWells)
plot(LDARocCurve, main = "ROC Curve: LDA", print.auc = T)



### QDA

library(MASS)
library(bestNormalize)

QDATrainWells = Wells[1:2520,]
QDATestWells = Wells[2521:3020,]


toScalearsenic = QDATrainWells$arsenic
toScaledist = QDATrainWells$dist

QDAarsenic = bestNormalize::boxcox(toScalearsenic)
QDAdist = bestNormalize::boxcox(toScaledist)

QDATrainWells$arsenic = QDAarsenic$x.t
QDATrainWells$dist = QDAdist$x.t

#Box Cox Transform for normality

QDATestWells$arsenic = as.vector(scale(((QDATestWells$arsenic^(-0.21) - 1)/-0.21)))
QDATestWells$dist = as.vector(scale(((QDATestWells$dist^(0.21) - 1)/0.21)))


QDAWellsModel = qda(switch~ arsenic + dist, data = QDATrainWells)



library(klaR)
partimat(factor(switch)~arsenic+dist, data=QDATrainWells, method="qda")

#Confusion Matrix
```

```r
QDApredictions = QDAWellsModel %>% predict(QDATestWells)

QDAConfusion = caret::confusionMatrix(QDApredictions$class, factor(QDATestWells$switch), positive = "1")

kable_styling(kable(QDAConfusion$table, caption = "Confusion Matrix : LDA",
                    col.names = c("Actual 0", "Actual 1")),latex_options = "HOLD_position")

#Construct ROC curve

QDATestWells$prob = QDApredictions$posterior[,2]
QDARocCurve <- roc(switch ~ prob, data = QDATestWells)
plot(QDARocCurve, main = "ROC Curve: QDA", print.auc = T)



### KNN K=1
library(DMwR)
KNNTrain = NewWells[1:2520,]
KNNTest = NewWells[2521:3020,]

# Scale variales
KNNTrain$logArsenic = as.vector(scale(KNNTrain$logArsenic))
KNNTrain$logDist = as.vector(scale(KNNTrain$logDist))
KNNTrain$Education = as.vector(scale(KNNTrain$Education))

KNNTest$logArsenic = as.vector(scale(KNNTest$logArsenic))
KNNTest$logDist = as.vector(scale(KNNTest$logDist))
KNNTest$Education = as.vector(scale(KNNTest$Education))

#Make predictions and plot confusion matrix
KNNpreds = kNN(Switch~ Education + Association +logArsenic +logDist, train = KNNTrain, test = KNNTest, n

kable_styling(kable(table(KNNpreds, KNNTest$Switch), caption = "Confusion Matrix: KNN, K=1",
                    col.names = c("Actual 0", "Actual 1")), latex_options = "HOLD_position")


### KNN K=1

#Make predictions and produce confusion matrix
KNNpreds5 = kNN(Switch~ Education + Association +logArsenic +logDist, train = KNNTrain, test = KNNTest,
                norm = F, k=5)
```

```r
kable_styling(kable(table(KNNpreds5, KNNTest$Switch), caption = "Confusion Matrix: KNN, K=5",
                    col.names = c("Actual 0", "Actual 1")), latex_options = "HOLD_position")



#Summarize Restults

SummaryDF = data.frame(Model = c("Logistic Regression", "LDA", "QDA", "KNN1", "KNN5"),
           ErrorRate = c(0.45, 0.47, 0.45, 0.45, 0.43),
           Sensitivity = c(0.84, 0.81, 0.825, 0.68, 0.76),
           Specificity = c(0.30, 0.28,  0.315, 0.44, 0.40),
           AUC = c(0.65, 0.637, 0.652, "", ""))

kable_styling(kable(SummaryDF, caption = "Comparison of Models"), latex_options = "HOLD_position")
```