Homework 7

Daria Barbour-Brown, Warren Kennedy, Bailey Ho

2023-05-16

Conceptual Problem

1. Linear discriminant analysis and logistic regression can both be used for binary classification. How would you decide to use one method versus the other in practice? Linear discriminant analysis assumes certain properties of the data, such as the normality and equality of variances in each class. If these assumptions are not me, LDA my not preform the way we need it to. Logistic regression makes fewer assumptions about the data distribution, making it a more flexible method to use when LDA assumptions are not met.

Logistic regression provides direct estimates of class membership probabilities. It gives us a good interpretation of the relationship between predictor variables and a binary outcome. From a logistic regression we may attain "odds ratios" and p-values, which we use to assess the impact of individual predictors. LDA does not directly estimate these probabilities but provides functions for classifying new observations. LDA focuses more on the separation between classes rather than individual predictors.

Lastly, when our sample size is large compared to the number of predictors, LDA performs well. In this case. LDA will more reliably estimate these parameters. On the other hand, logistic regression is better suited for smaller sample sizes with flexibility to deal with higher numbers of predictors. If you are dealing with high-dimensional data with more predictors than observations, logistic regression is considered to be more appropriate.

Application Problems

2. Fitting a logistic regression model with continuous covariate and categorical covariate.

```
library(mlbench)
library(ggplot2)
library(dplyr)
```

(a) Fit a logistic regression with diabetes (Y/N) as the response, and glucose levels as a covariate, as well as an indicator for an individual having a BMI greater than 35 (BMI is given by mass in this dataset). Print the summary of this regression object.

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

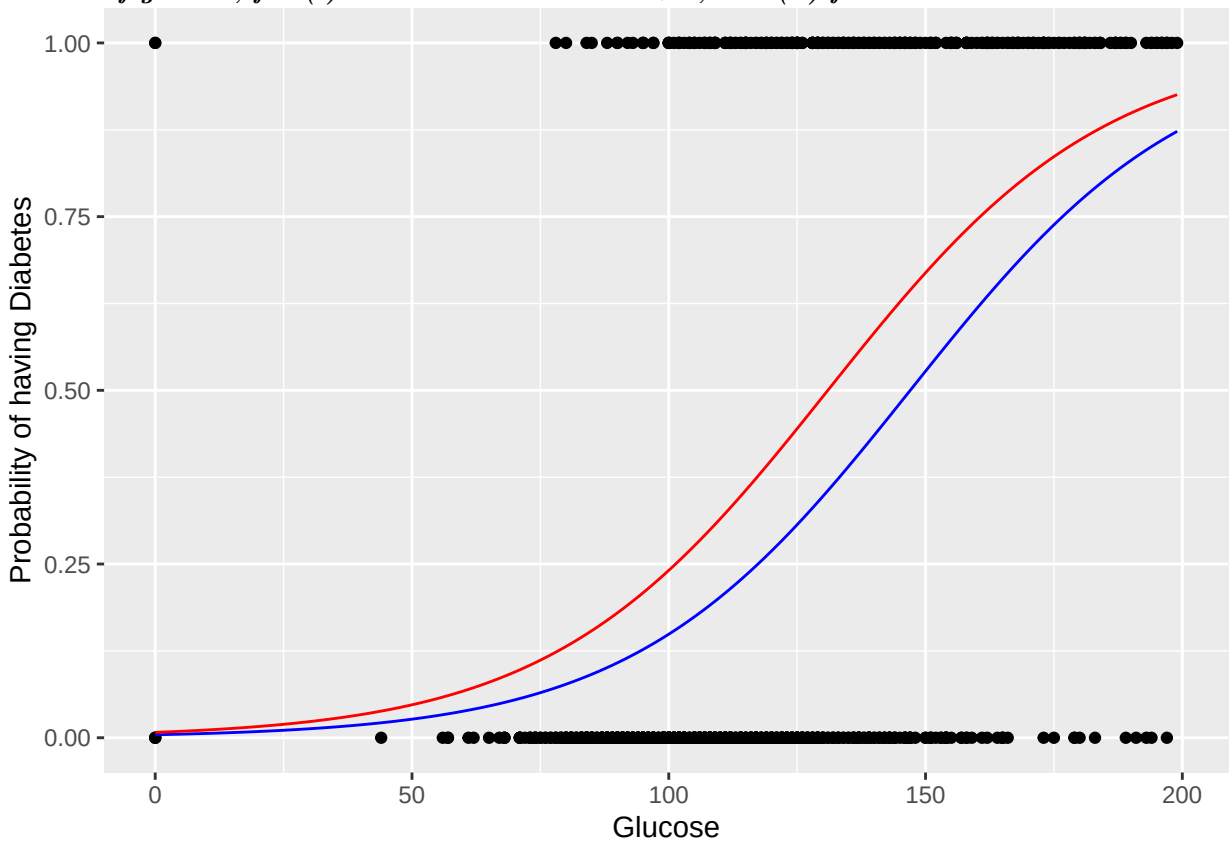
```
data("PimaIndiansDiabetes") #loads data under "name"
pimaData <- PimaIndiansDiabetes
#create new numeric col of all 1s
pimaData$response <- 1
#crate response col, transform string to categorical
pimaData$response[pimaData$diabetes == 'neg'] <- 0
#bmi > 35 col
pimaData$bmi <- 0
pimaData$bmi[pimaData$mass > 35] <- 1

logRegMulti <- glm(response~glucose + bmi, data=pimaData, family="binomial")
summary(logRegMulti)
```

```
##
## Call:
## glm(formula = response ~ glucose + bmi, family = "binomial",
##     data = pimaData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2489  -0.7792  -0.5215   0.8596   3.1177
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.44651    0.42633 -12.775  < 2e-16 ***
## glucose      0.03705    0.00327  11.330  < 2e-16 ***
## bmi          0.59434    0.18242   3.258  0.00112 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 993.48  on 767  degrees of freedom
## Residual deviance: 798.17  on 765  degrees of freedom
## AIC: 804.17
##
## Number of Fisher Scoring iterations: 4
```

```
#Glucose for Low BMI
#       vector of 1s for intercept    vector of random glucose levels btw max and min        all bmi's
BMILow <- cbind(rep(1,10000), seq(from=min(pimaData$glucose),to=max(pimaData$glucose),length.out=1000),
lowPredxb <- BMILow%*%coef(logRegMulti)
lowPreds <-  1/(1+exp(-lowPredxb))
lowPredDF <- data.frame(x=BMILow[,2], preds=lowPreds)

#Glucose for High BMI
```

```
BMIHigh <- cbind(rep(1,10000), seq(from=min(pimaData$glucose),to=max(pimaData$glucose),length.out=1000)
highPredxb <- BMIHigh%*%coef(logRegMulti)
highPreds <-  1/(1+exp(-highPredxb))
highPredDF <- data.frame(x=BMIHigh[,2], preds=highPreds)

ggplot() +
  geom_point(data=pimaData,aes(y=response, x=glucose)) +
  geom_line(data=lowPredDF,aes(x=x,y=lowPreds),color="Blue") +
  geom_line(data=highPredDF,aes(x=x,y=highPreds),color="Red") +
  labs(y="Probability of having Diabetes", x="Glucose")
```

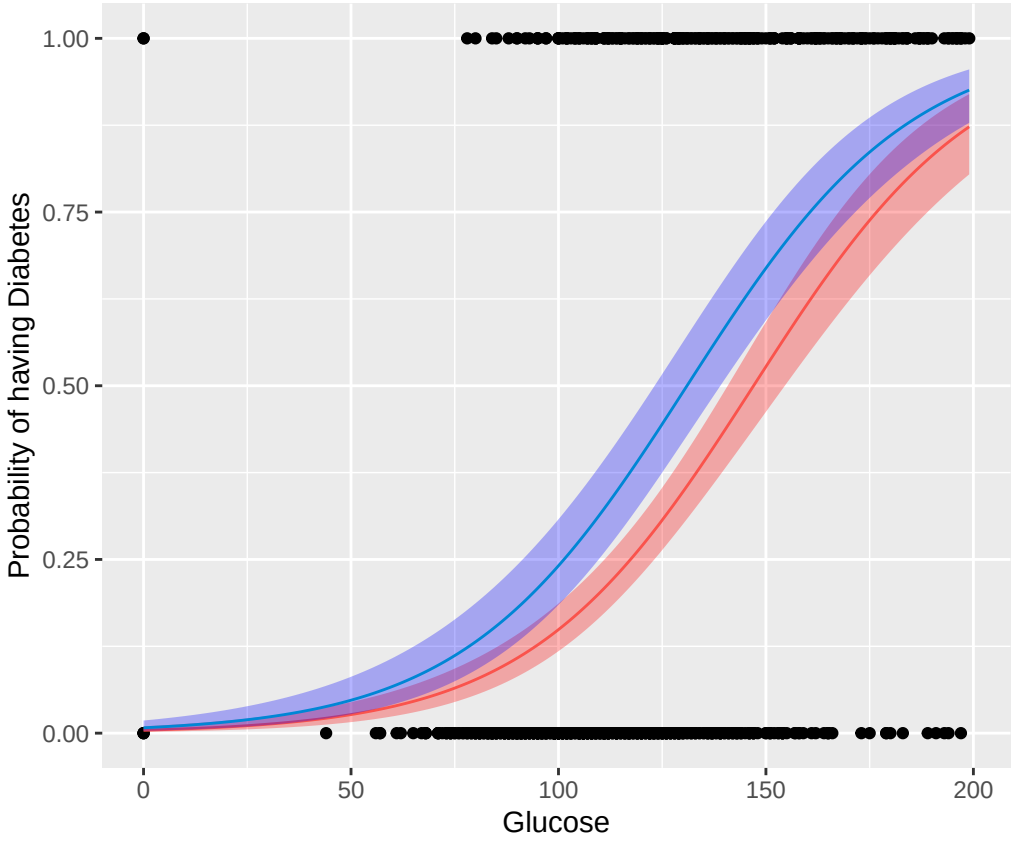
(b) Plot glucose levels versus diabetes (Y/N). Add the predicted probability of having diabetes as a function of glucose, for (i) individuals with BMI <35, and (ii) for individuals with BMI >35.




```
#CI Low BMI
lowSexb <- sqrt(diag(BMILow%*%vcov(logRegMulti)%*%t(BMILow)))
lowCixb <- cbind(lowPredxb-1.96*lowSexb,lowPredxb+1.96*lowSexb)
ciLow <- 1/(1+exp(-lowCixb))
lowPredDF <- lowPredDF %>% mutate(lb=ciLow[,1],ub=ciLow[,2])

#CI High BMI
highSexb <- sqrt(diag(BMIHigh%*%vcov(logRegMulti)%*%t(BMIHigh)))
highCixb <- cbind(highPredxb-1.96*highSexb,highPredxb+1.96*highSexb)
ciHigh <- 1/(1+exp(-highCixb))
```

```
highPredDF <- highPredDF %>% mutate(lb=ciHigh[,1],ub=ciHigh[,2])

ggplot() +
  geom_point(data=pimaData,aes(y=response, x=glucose)) +
  geom_line(data=lowPredDF,aes(x=x,y=lowPreds, color="BMI < 35")) +
  geom_ribbon(data=lowPredDF,aes(x=x,ymin=lb,ymax=ub),alpha=0.3,fill="Red") +
  geom_line(data=highPredDF,aes(x=x,y=highPreds, color="BMI > 35")) +
  geom_ribbon(data=highPredDF,aes(x=x,ymin=lb,ymax=ub),alpha=0.3,fill="Blue") +
  labs(y="Probability of having Diabetes", x="Glucose")
```

(c) Add 95% confidence intervals for the predicted probability of having diabetes to your plots for both predictions in (b).



3. LDA

```
library(MASS)
```

(a) Split the data into training/testing sets, with 80% training. Fit a LDA model with diabetes (Y/N) as the response and glucose and BMI as covariates (use the actual BMI values this time.)

```
##
## Attaching package: 'MASS'
```

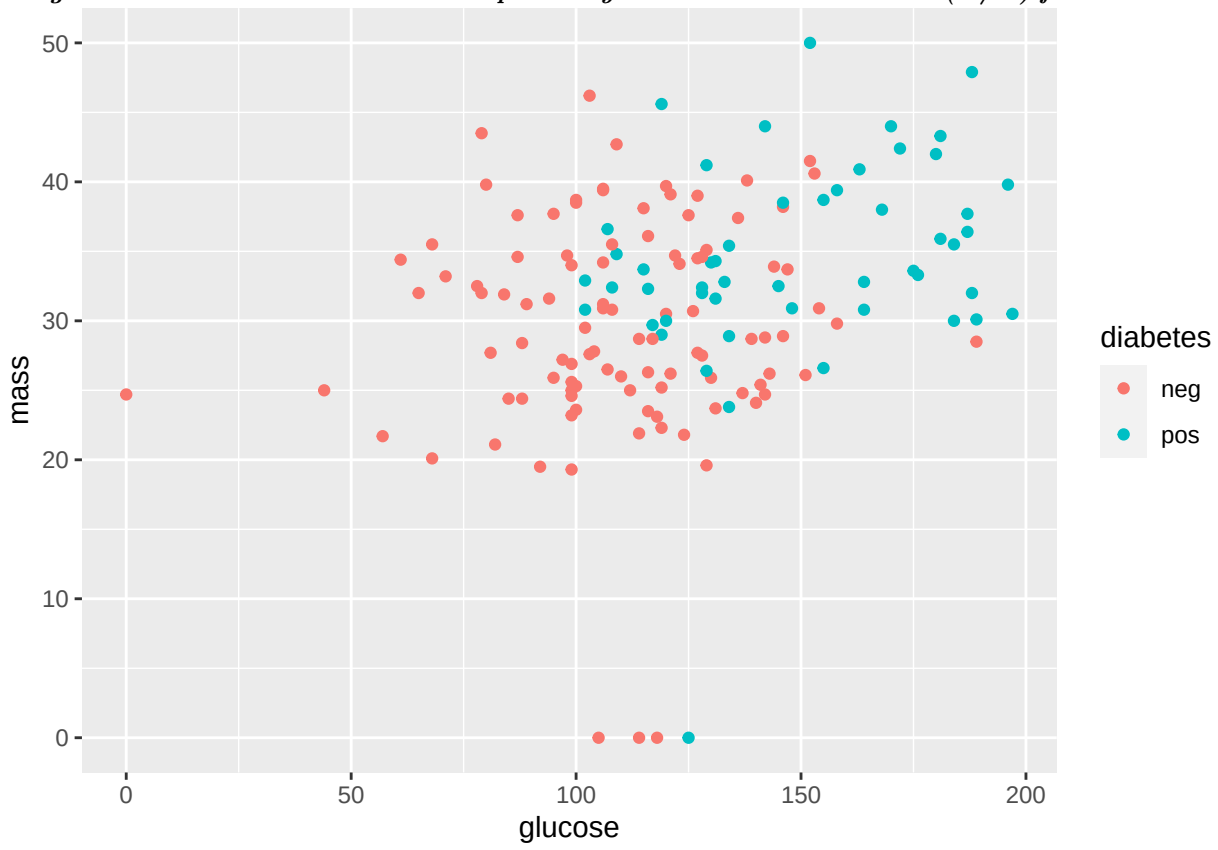
```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
set.seed(1)
train = sample(768, 768*.8)
train_set <- pimaData[train, ]
test_set <- pimaData[-train, ]

lda_fit <- lda(response ~ glucose + mass, data = pimaData,
    subset = train)
lda_fit
```

```
## Call:
## lda(response ~ glucose + mass, data = pimaData, subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6465798 0.3534202
##
## Group means:
##    glucose     mass
## 0 109.9547 30.51411
## 1 139.4562 35.29908
##
## Coefficients of linear discriminants:
##                LD1
## glucose 0.02960219
## mass    0.06257069
```

```

ggplot(test_set, aes(x=glucose,y=mass)) + geom_point(aes(color = diabetes))
```

(b) Plot glucose versus BMI and color the points by their true diabetes class (Y/N) for the test set.




```
Diabetic <- test_set$response == 1
NonDiabetic <- test_set$response == 0

Diabetic.df <- test_set[Diabetic, ]
NonDiabetic.df <- test_set[NonDiabetic, ]

diabetic.gluc.mean <- mean(Diabetic.df$glucose)
diabetic.mass.mean <- mean(Diabetic.df$mass)

nondiabetic.gluc.mean <- mean(NonDiabetic.df$glucose)
nondiabetic.mass.mean <- mean(NonDiabetic.df$mass)

x1 <- cbind(diabetic.gluc.mean,diabetic.mass.mean)
x2 <- cbind(nondiabetic.gluc.mean,nondiabetic.mass.mean)

pi1 = 0.6465798
pi2 = 0.3534202

mu1hat <- colMeans(x1)
mu2hat <- colMeans(x2)
Sigmahat <- cov(rbind(x1,x2))
intercept <- c(log(pi1/pi2) - 0.5*t(mu1hat+mu2hat)%*%solve(Sigmahat, tol = 1e-18)%*%(mu1hat-mu2hat))
slope <- solve(Sigmahat, tol = 1e-18)%*%(mu1hat-mu2hat)
```

```
b0 <- intercept/(-slope[2])
b1 <- slope[1]/(-slope[2])
```

(c) Add the LDA decision boundary for the model fit in (a).

```
lda_pred <- predict(lda_fit, newdata = test_set)
temp_table <- table(lda_pred$class, test_set$response)
temp_table
```

(d) Print the 2x2 confusion matrix of the predictions on the test set versus their true values. Comment on the performance of LDA for predicting diabetes using glucose and BMI.

```
##
##      0  1
##   0 93 22
##   1 10 29
```

```
#Calculating accuracy
mean(lda_pred$class == test_set$response)
```

```
## [1] 0.7922078
```

The LDA's predictions are accurate almost 80% of the time.

4. QDA

```
qda_fit <- qda(response ~ glucose + mass, data = pimaData,
    subset = train)
qda_fit
```

(a) Using the same train/test split as in 2a, fit a QDA with the same variables.

```
## Call:
## qda(response ~ glucose + mass, data = pimaData, subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.6465798 0.3534202
##
## Group means:
##    glucose     mass
## 0 109.9547 30.51411
## 1 139.4562 35.29908
```

```
# OLD CODE
#temp_table <- table(Wage$hi,round(predict(logReg,type='response')))
#temp_table

# NEW CODE
qda_pred <- predict(qda_fit, newdata = test_set)
temp_table2 <- table(qda_pred$class, test_set$response)
temp_table2
```

(b) Print the 2x2 confusion matrix of the predictions on the test set versus their true values.

```
##
##      0  1
##   0 94 23
##   1  9 28
```

```
mean(qda_pred$class == test_set$response)
```

(c) Compare the predictions of LDA versus QDA. Which model would you recommend in this case?

```
## [1] 0.7922078
```

Alternatively:

If we calculate accuracy for QDA: (TP + TN) / (TP + TN + FP + FN) = (28 + 94) / (28 + 94 + 23 + 9) = 0.7922078

If we calculate accuracy for LDA: (TP + TN) / (TP + TN + FP + FN) = (29 + 93) / (29 + 93 + 22 + 10) = 0.7922078

We conclude that both LDA and QDA can make correct predictions about diabetes (using BMI and glucose levels) about 80% of the time.