A4

Yuqi Gu

7/23/2024

```
setwd('/Users/yuki/Desktop/STAT 341/Assignments/A4')
votes <- read.csv("votes.csv")
```

QUESTION 1: Randomization Tests

(a)

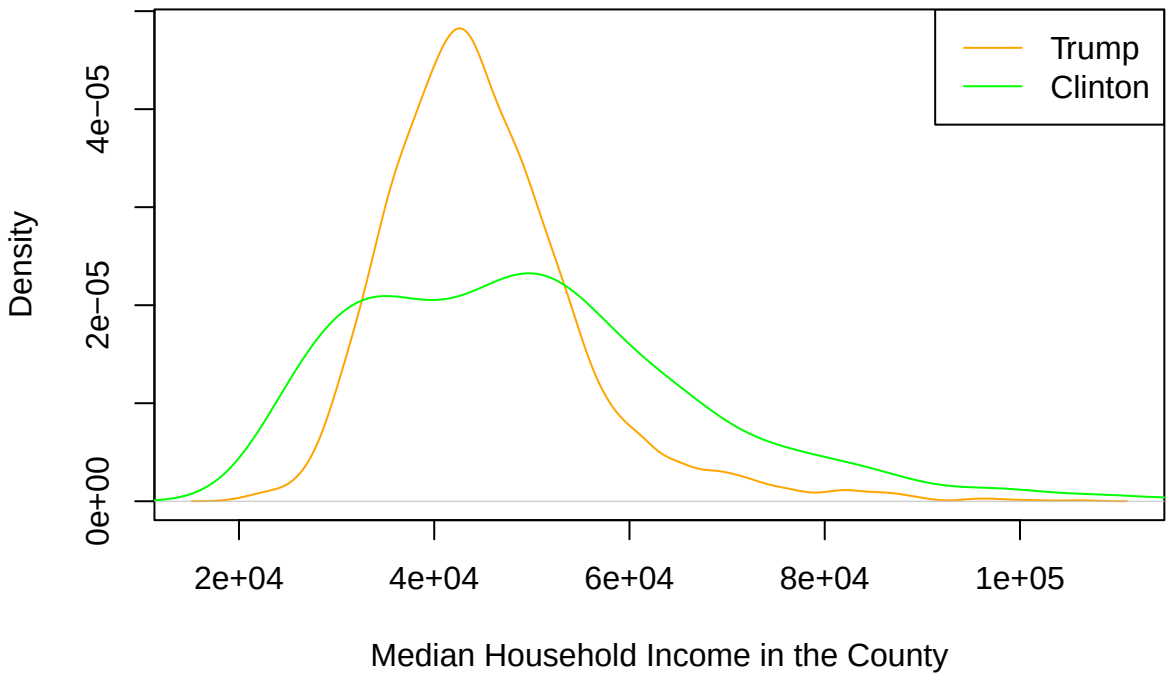
```
votes$winner <- ifelse(votes$votes_gop_2016 > votes$votes_dem_2016,
                       "Trump", "Clinton")
mean(votes$winner == "Trump")
```

```
## [1] 0.8431877
```

The proportion of counties in which Trump won is 0.8431877.

(b)

```
Tmi <- votes$INC110213[votes$winner == "Trump"]
Cmi <- votes$INC110213[votes$winner == "Clinton"]

plot(density (Tmi), col = "orange",
     xlab = "Median Household Income in the County",
     main = "Distributions of County-Median-Income by Winning Candidate")
lines(density (Cmi), col = "green")
legend("topright", legend = c("Trump", "Clinton"), lty=1, col = c("orange", "green"))
```




```
mad <- function(y){
  return( median(abs(y - median(y))) )
}

bskew <- function(y){
  Q1 <- quantile(y, 0.25)
  Q2 <- quantile(y, 0.50)
  Q3 <- quantile(y, 0.75)
  return( as.numeric((Q3 - 2*Q2 + Q1)/(Q3 - Q1)))
}

median(Tmi)
```

```
## [1] 43764.5
```

```
median(Cmi)
```

```
## [1] 48224.5
```

```
mad(Tmi)
```

```
## [1] 5757.5
```

```
mad(Cmi)
```

```
## [1] 11698
```

```
bskew(Tmi)
```

```
## [1] 0.07041582
```

```
bskew(Cmi)
```

```
## [1] -0.09081773
```

Based on the above plot, we observe that the distribution of county-median-income by Trump is generally more centered or concentrated. On the contrary, the distribution of county-median-income by Clinton is generally less centered. There are more variability of the distribution of county-median-income for Clinton than for Trump. The peak of the distribution of county-median-income for Trump is higher than that for Clinton. Thus, the distribution of county-median-income by Trump looks different from that by Clinton. The shapes of these two distributions seem different.

The median of these two distribution is different, with the center of the median income of the counties in which Trump won is 43764.5 and it is 48224.5 for the counites in which Clinton won. Furthermore, The spread of these two distributions are different. The counties in which Trump won have less variability in median incomes, while counties in which Clinton won show a wider range of incomes. Specifically, the MAD of the median household income in the counties for Trump is 5757.5 and it is 11698 for Clinton. The distribution of the median income of the counties in which Trump won is less spread than Clinton.Both distributions are a little bit skewed. The distribution of the median household income in the counties for Trump has skewness of 0.07041582, indicating that it is right-skewed. On the contrary, the distribution of the median household income in the counties for Clinton is -0.09081773, indicating that it is left-skewed. But the values of these two skewness are approximately zero, so the skewness of these two distributions are not obvious.

(c)

$H_0 : \mathcal{P}_C$ *and* $\mathcal{P}_T$ *are drawn randomly from the same population of the median household income in the county in* 2013.

(d) i.

```
D1 <- function(pop){
  abs(median(pop[[1]]$INC110213)-median(pop[[2]]$INC110213))
}
```

ii.

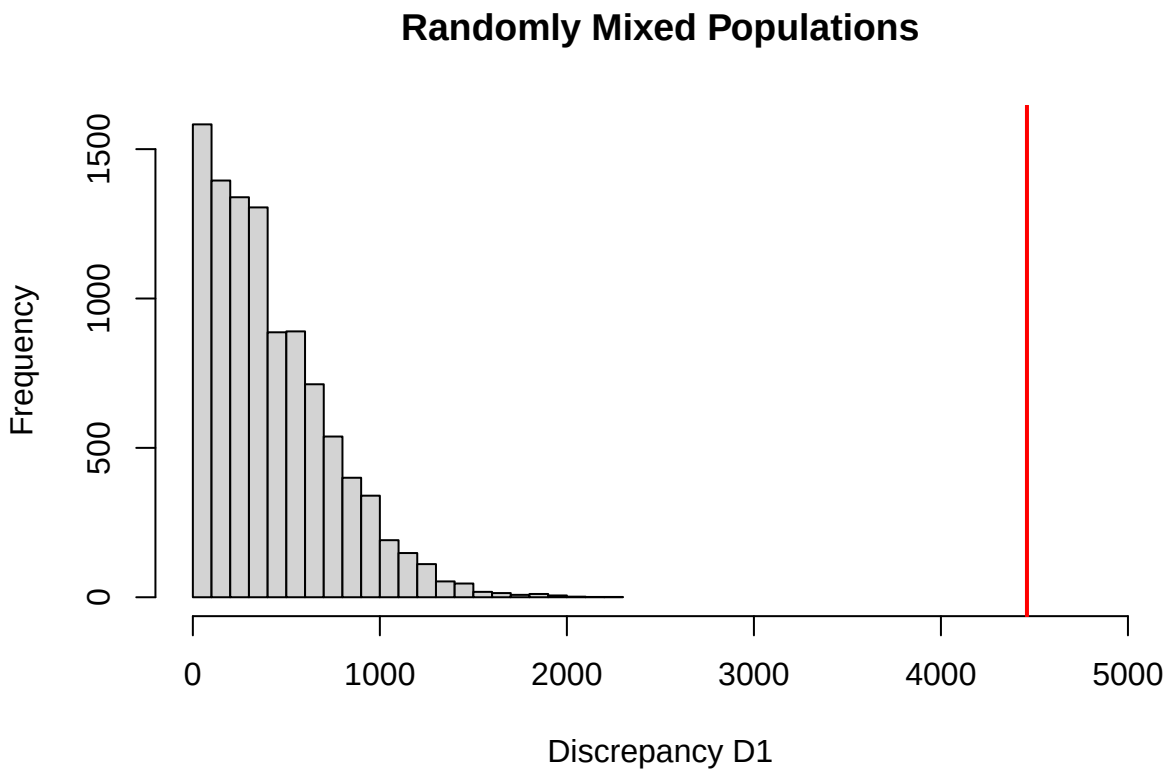
```
pop <- list( pop1 = votes[votes$winner == "Trump", ],
             pop2 = votes[votes$winner == "Clinton", ] )
D1(pop)
```

```
## [1] 4460
```

The observed discrepancy using the function defined in part i. is 4460. The absolute difference between medians of pop1 and pop2 is 4460.

iii.

```
mixRandomly <- function(pop) {
    pop1 <- pop$pop1
    n_pop1 <- nrow(pop1)

    pop2 <- pop$pop2
    n_pop2 <- nrow(pop2)

    mix <- rbind(pop1, pop2)
    select4pop1 <- sample(1:(n_pop1 + n_pop2), n_pop1, replace = FALSE)

    new_pop1 <- mix[select4pop1, ]
    new_pop2 <- mix[-select4pop1, ]
    list(pop1 = new_pop1, pop2 = new_pop2)
}

set.seed(341)
diffincomes <- sapply(1:10000, FUN = function(...) {
    D1(mixRandomly(pop))
})

hist(diffincomes, breaks = 20, main = "Randomly Mixed Populations",
     xlim = c(0, 5000), xlab = "Discrepancy D1", col = "lightgrey")
abline(v = D1(pop), col = "red", lwd = 2)
```



iv.

```
mean(diffincomes >= D1(pop))
```

```
## [1] 0
```

The p-value associated with this test is 0. With a p-value of 0, we have very strong evidence against the null hypothesis that $\mathcal{P}_C$ *and* $\mathcal{P}_T$ *are drawn randomly from the same population of the median household income in the county in*

(e) i.

```
D2 <- function(pop){
  abs(mad(pop[[1]]$INC110213)/mad(pop[[2]]$INC11021)-1)
}
```

ii.

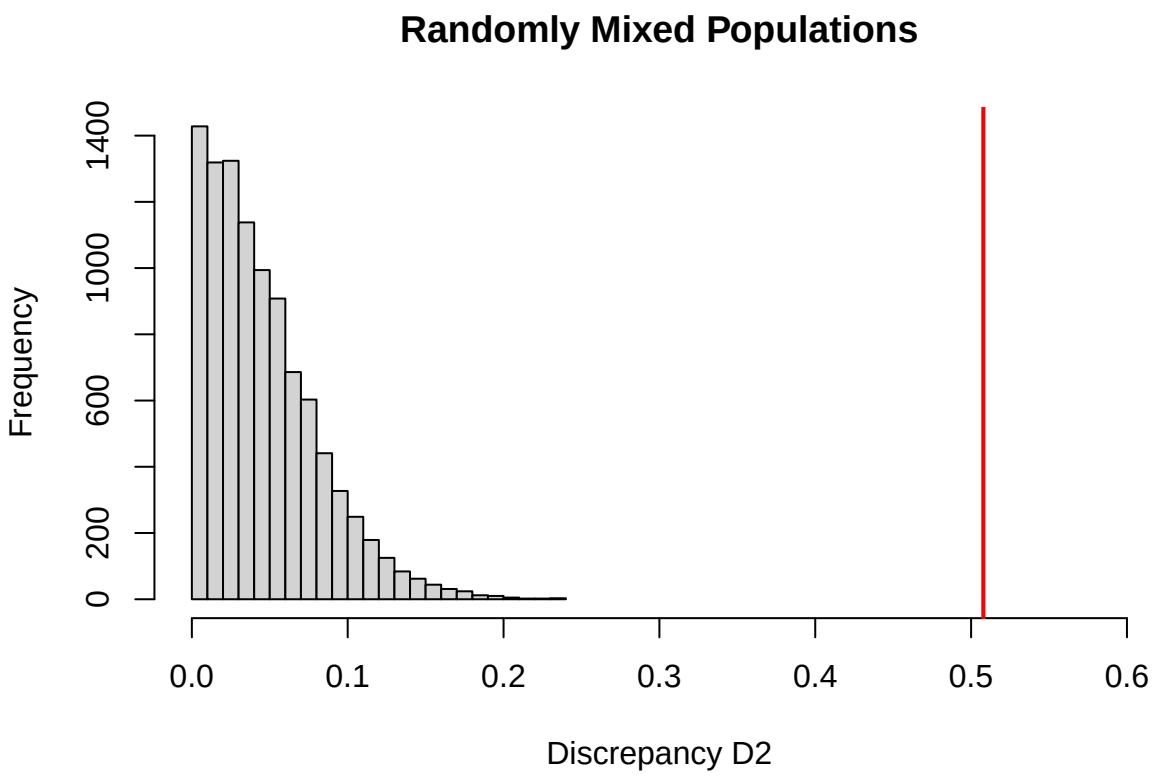
```
D2(pop)
```

```
## [1] 0.5078218
```

The observed discrepancy using the function defined in part i. is 0.5078218.

iii.

```
set.seed(341)
getd2 <- sapply(1:10000, FUN = function(...) {
    D2(mixRandomly(pop))
})

hist(getd2, breaks = 20, main = "Randomly Mixed Populations",
     xlim = c(0, 0.6), xlab = "Discrepancy D2", col = "lightgrey")
abline(v = D2(pop), col = "red", lwd = 2)
```



iv.

```
mean(getd2 >= D2(pop))
```

```
## [1] 0
```

The p-value associated with this test is 0. With a p-value of 0, we have very strong evidence against the null hypothesis that $\mathcal{P}_C$ *and* $\mathcal{P}_T$ *are drawn randomly from the same population of the median household income in the county ir*

(f) i.

```
D3 <- function(pop){
  abs(bskew(pop[[1]]$INC110213)-bskew(pop[[2]]$INC110213))
}
```

ii.

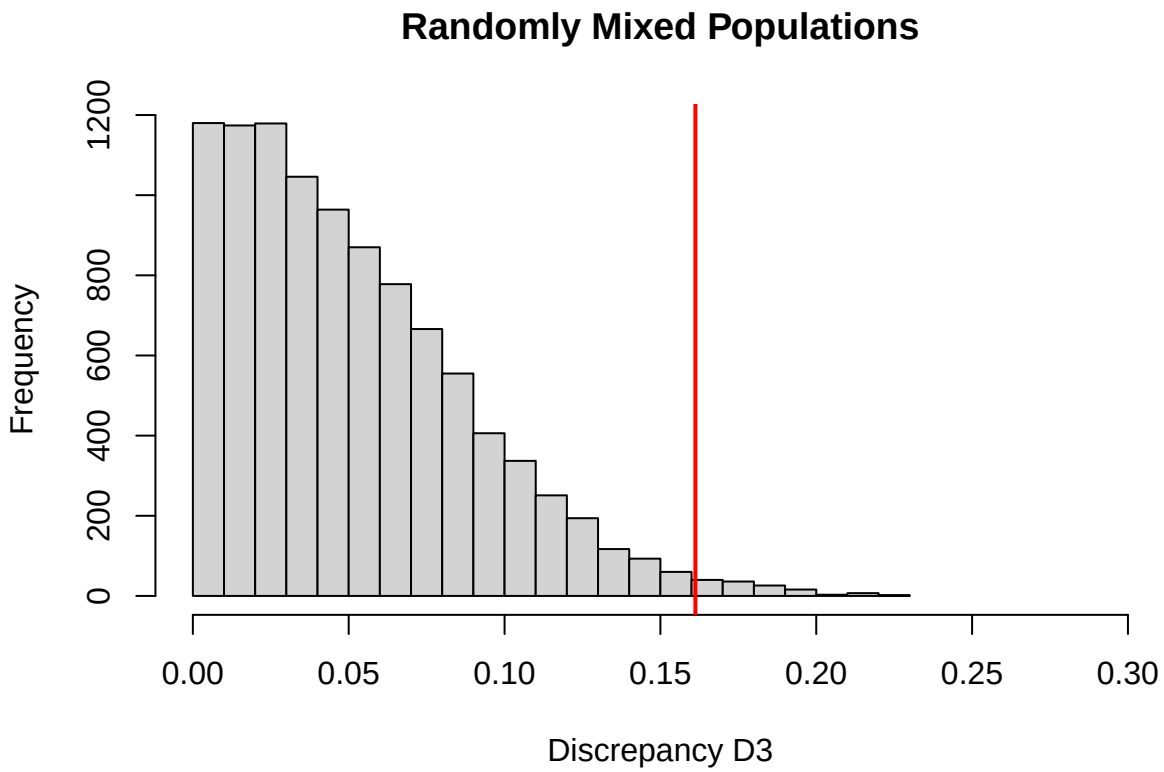
```
D3(pop)
```

```
## [1] 0.1612335
```

The observed discrepancy using the function defined in part i. is 0.1612335.

iii.

```
set.seed(341)
getd3 <- sapply(1:10000, FUN = function(...) {
    D3(mixRandomly(pop))
})

hist(getd3, breaks = 20, main = "Randomly Mixed Populations",
     xlim = c(0, 0.3), xlab = "Discrepancy D3", col = "lightgrey")
abline(v = D3(pop), col = "red", lwd = 2)
```



iv.

```
mean(getd3 >= D3(pop))
```

```
## [1] 0.0122
```

The p-value associated with this test is 0.0122. With a p-value of 0.0122, we have evidence against the null hypothesis that $\mathcal{P}_C$ *and* $\mathcal{P}_T$ *are drawn randomly from the same population of the median household income in the county in* 2

(g)

```
calculatePVmulti <- function(pop, discrepancies, M_outer, M_inner) {

    ## Local function to calculate the p-values over the discrepancies and
    ## return their minimum

    getPVmin <- function(basePop, discrepancies, M) {
        observedVals <- sapply(discrepancies, FUN = function(discrepancy) {
            discrepancy(basePop)
        })

        K <- length(discrepancies)

        total <- Reduce(function(counts, i) {
            # mixRandomly mixes the two populations randomly, so the new
            # sub-populations are indistinguishable
            NewPop <- mixRandomly(basePop)

            ## calculate the discrepancy and counts
            Map(function(k) {
                Dk <- discrepancies[[k]](NewPop)
                if (Dk >= observedVals[k])
                  counts[k] <<- counts[k] + 1
            }, 1:K)
            counts
        }, 1:M, init = numeric(length = K))

        PVs <- total/M
        min(PVs)
    }

    PVmin <- getPVmin(pop, discrepancies, M_inner)

    total <- Reduce(function(count, m) {
        basePop <- mixRandomly(pop)
        if (getPVmin(basePop, discrepancies, M_inner) <= PVmin)
            count + 1 else count
    }, 1:M_outer, init = 0)

    PVstar <- total/M_outer
    return(PVstar)
}
```

i.

```
discrepancies <- list(D1, D2, D3)
set.seed(341)
M_inner <- 100
M_outer <- 1000
calculatePVmulti(pop, discrepancies, M_outer, M_inner)
```

```
## [1] 0.03
```

ii.

The p-value* is 0.03, which is smaller than 0.05. Thus, we reject the null hypothesis that $\mathcal{P}_C$ *and* $\mathcal{P}_T$ *are drawn randomly from* Thus, these two populations are randomly drawn from different population of the median household income in the county in 2013. By referring to the density plot in part (b), this conclusion is not surprising since we observe that the shape of these two distributions are not similar.

(h)

Multiple testing problem exists when we try to consider and calculate several discrepancy measures at the same time when doing the hypothesis test. In this question, we tried to use three discrepancy measures to test the null hypothesis of part (c). This makes the probability of yielding a Type I error inflated. Thus, it is better to use a more customized approach that just integrates the information gathered from each of these three discrepancy measures instead of evaluating them separately. This means that we will just get one p-value that will be used to compare with the significance level and derive the conclusions. This will help us to reduce the probability of making Type I error and protect from the multiple testing problem. Thus, the approach taken in part (g) is to be preferred to considering three separate tests based on D_1, D_2, and D_3 individually.

QUESTION 2: Newton-Raphson Logistic Regression

(a)

$$\frac{\partial l(\alpha,\beta;\mathcal{P})}{\partial \alpha} = \sum_{u\in\mathcal{P}} y_u - \sum_{u\in\mathcal{P}} \frac{e^{\alpha+\beta x_u}}{1+e^{\alpha+\beta x_u}}$$

$$\frac{\partial l(\alpha,\beta;\mathcal{P})}{\partial \beta} = \sum_{u\in\mathcal{P}} y_u x_u - \sum_{u\in\mathcal{P}} \frac{x_u e^{\alpha+\beta x_u}}{1+e^{\alpha+\beta x_u}}$$

(b)

$$\begin{aligned}\frac{\partial^2 l(\alpha,\beta;\mathcal{P})}{\partial \alpha^2} &= -\sum_{u\in\mathcal{P}} \frac{e^{\alpha+\beta x_u}(1+e^{\alpha+\beta x_u}) - (e^{\alpha+\beta x_u})(e^{\alpha+\beta x_u})}{(1+e^{\alpha+\beta x_u})^2} \\ &= -\sum_{u\in\mathcal{P}} \frac{e^{\alpha+\beta x_u}}{1+e^{\alpha+\beta x_u}} + \sum_{u\in\mathcal{P}} \frac{e^{2\alpha+2\beta x_u}}{(1+e^{\alpha+\beta x_u})^2}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 l(\alpha,\beta;\mathcal{P})}{\partial \alpha\partial\beta} = \frac{\partial^2 l(\alpha,\beta;\mathcal{P})}{\partial \beta\partial\alpha} &= -\sum_{u\in\mathcal{P}} \frac{x_u e^{\alpha+\beta x_u}(1+e^{\alpha+\beta x_u}) - (e^{\alpha+\beta x_u})(e^{\alpha+\beta x_u} x_u)}{(1+e^{\alpha+\beta x_u})^2} \\ &= -\sum_{u\in\mathcal{P}} \frac{x_u e^{\alpha+\beta x_u}}{1+e^{\alpha+\beta x_u}} + \sum_{u\in\mathcal{P}} \frac{e^{2\alpha+2\beta x_u} x_u}{(1+e^{\alpha+\beta x_u})^2}\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 l(\alpha,\beta;\mathcal{P})}{\partial \beta^2} &= -\sum_{u\in\mathcal{P}} \frac{(x_u)^2 e^{\alpha+\beta x_u}(1+e^{\alpha+\beta x_u}) - (e^{\alpha+\beta x_u})(e^{\alpha+\beta x_u})(x_u)^2}{(1+e^{\alpha+\beta x_u})^2} \\ &= -\sum_{u\in\mathcal{P}} \frac{(x_u)^2 e^{\alpha+\beta x_u}}{1+e^{\alpha+\beta x_u}} + \sum_{u\in\mathcal{P}} \frac{e^{2\alpha+2\beta x_u}(x_u)^2}{(1+e^{\alpha+\beta x_u})^2}\end{aligned}$$

(c)

```
createLogisticPsiFns <- function(x, y) {
  psiFn <- function(theta) {
    alpha = theta[1]
    beta = theta[2]
    firstderivative_alpha <- sum(y) - sum(exp(alpha + beta * x) / (1 + exp(alpha + beta * x)))
    firstderivative_beta <- sum(y*x) - sum(x*exp(alpha + beta * x) / (1 + exp(alpha + beta * x)))
    c(firstderivative_alpha, firstderivative_beta)
  }

  psiPrimeFn <- function(theta) {
    alpha = theta[1]
    beta = theta[2]
    psiprime <- matrix(0, nrow = 2, ncol = 2)

    psiprime[1,1] <- -sum(exp(alpha + beta* x) / (1+exp(alpha + beta * x))) +
                      sum(exp(2 * alpha + 2 * beta * x) / (1 + exp(alpha + beta * x))^2)

    psiprime[1,2] <- -sum(x * exp(alpha + beta* x) / (1 +exp(alpha + beta * x))) +
                      sum(exp(2 * alpha + 2 * beta * x) * x / (1 + exp(alpha + beta * x))^2)
```

```
    psiprime[2,1] <- psiprime[1,2]
    psiprime[2,2] <- -sum(x^2 * exp(alpha + beta* x) / (1+ exp(alpha + beta * x))) +
                      sum(exp(2 * alpha + 2 * beta * x)*x^2 / (1 + exp(alpha + beta * x))^2)
    return(psiprime)
  }

  list(psiFn = psiFn, psiPrimeFn = psiPrimeFn)
}
```

(d)

```
NewtonRaphson <- function(theta, psiFn, psiPrimeFn, dim, testConvergenceFn = testConvergence,
    maxIterations = 100, tolerance = 1e-06, relative = FALSE) {
    if (missing(theta)) {
        ## need to figure out the dimensionality
        if (missing(dim)) {
            dim <- length(psiFn())
        }
        theta <- rep(0, dim)
    }
    converged <- FALSE
    i <- 0
    while (!converged & i <= maxIterations) {
        thetaNew <- theta - solve(psiPrimeFn(theta), psiFn(theta))
        converged <- testConvergenceFn(thetaNew, theta, tolerance = tolerance,
            relative = relative)

        theta <- thetaNew
        i <- i + 1
    }
    ## Return last value and whether converged or not
    list(theta = theta, converged = converged, iteration = i, fnValue = psiFn(theta))
}

testConvergence <- function(thetaNew, thetaOld, tolerance = 1e-10, relative = FALSE) {
    sum(abs(thetaNew - thetaOld)) < if (relative)
        tolerance * sum(abs(thetaOld)) else tolerance
}
```

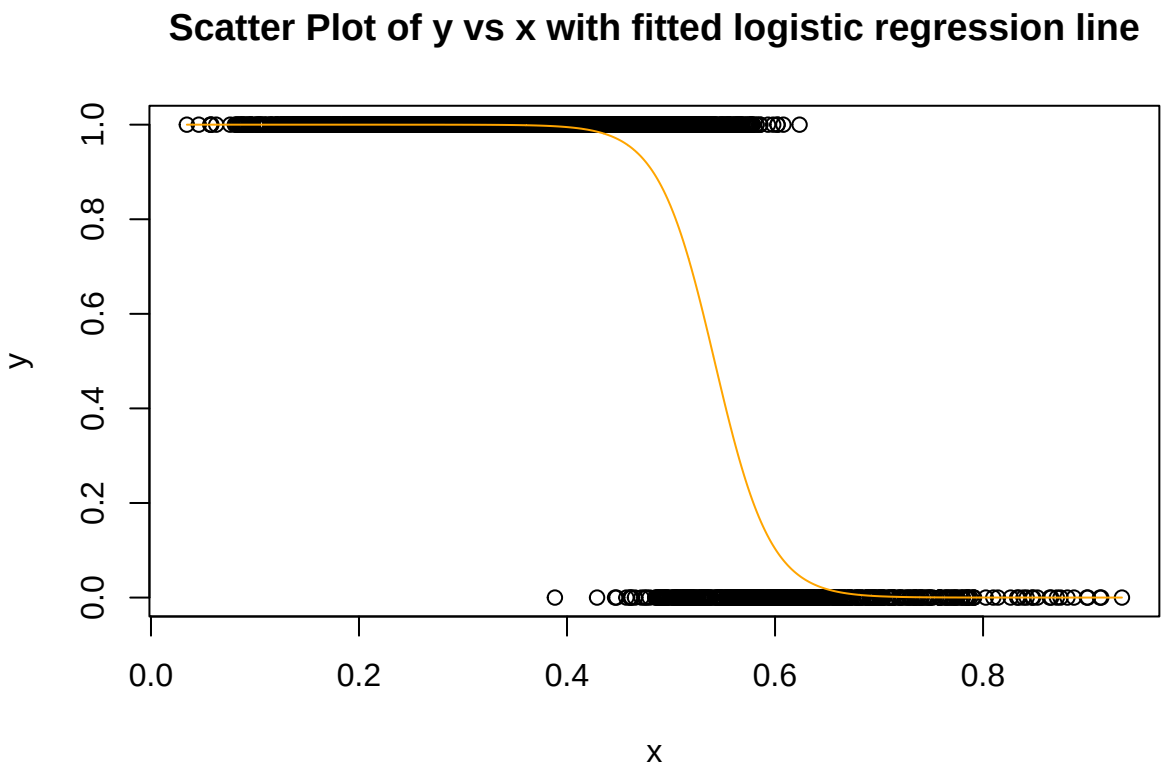
```
y = (votes$winner == "Trump")*1
x = votes$Obama

psiFns <- createLogisticPsiFns(x,y)
alpha0 <- log(mean(y)/(1-mean(y)))
beta0 <- 0
output <- NewtonRaphson(theta = c(alpha0, beta0), psiFn = psiFns$psiFn, psiPrimeFn = psiFns$psiPrimeFn)
output
```

```
## $theta
## [1]  20.12942 -37.14827
##
```

```
## $converged
## [1] TRUE
##
## $iteration
## [1] 9
##
## $fnValue
## [1] 0 0
```

(e)

```
p_hat <- function(x, theta) {
  alpha_hat <- theta[1]
  beta_hat <- theta[2]
  exp(alpha_hat + beta_hat*x) / (1+exp(alpha_hat + beta_hat*x))
}

plot(x, y, main = "Scatter Plot of y vs x with fitted logistic regression line")
lines(sort(x), p_hat(sort(x), output$theta), col = "orange")
```



This model is a good fit since the orange line, which is the fitted logistic regression line, is close to the black points as derived by the values of y and x. In other words, the trend of the scatter plot of y versus x is similar as the fitted logistic regression line.

(f)

```
pval <- p_hat(x, output$theta)
c <- 0.5
yh <- (pval >= c)*1
confusion_matrix <- table(y, yh)
confusion_matrix
```

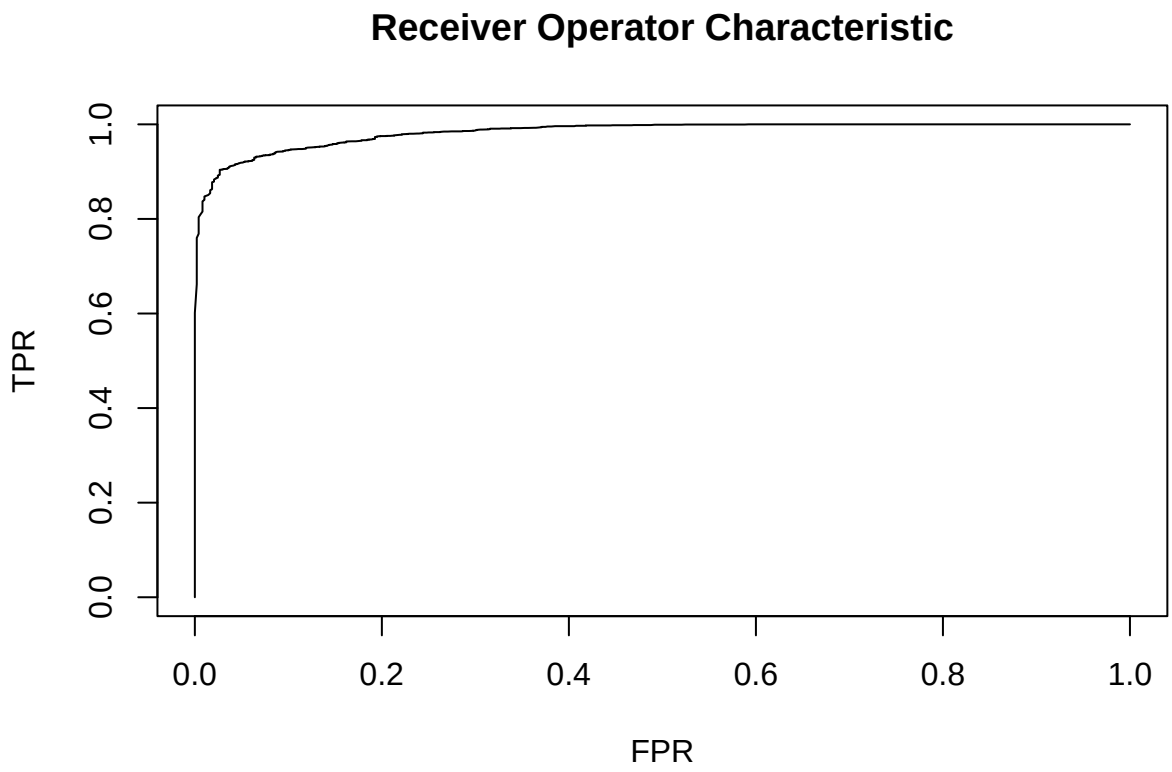
```
##    yh
## y      0    1
##   0  394   94
##   1   70 2554
```

```
TN <- confusion_matrix[1, 1]
TP <- confusion_matrix[2, 2]
FP <- confusion_matrix[1, 2]
FN <- confusion_matrix[2, 1]
accuracy <- (TN + TP)/(TN + FP + FN + TP)
accuracy
```

```
## [1] 0.9473008
```

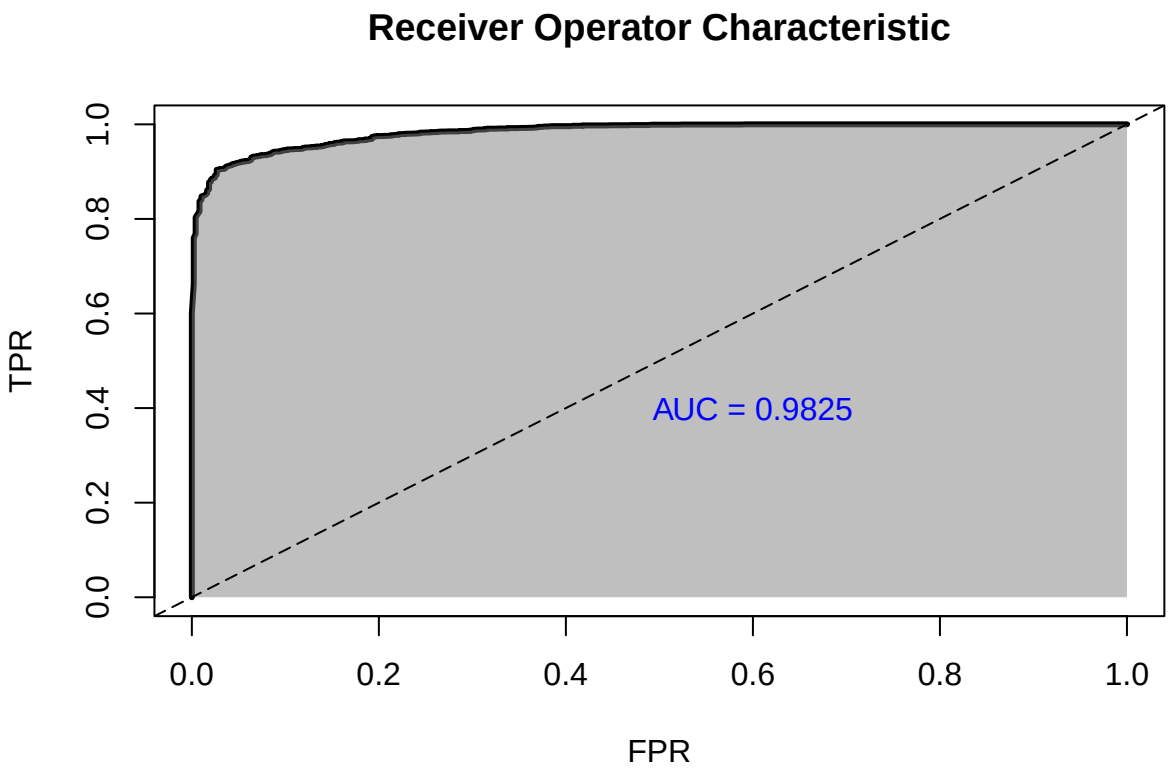
(g) i.

```
c = seq(from=0, to=1, length.out=500)
TPRvals <- rep(0, length(c))
FPRvals <- rep(0, length(c))

iter = 1
for (i in c){
  yh <- (pval >= i)*1
  TN <- sum((yh == 0) & (y == 0))
  TP <- sum((yh == 1) & (y == 1))
  FP <- sum((yh == 1) & (y == 0))
  FN <- sum((yh == 0) & (y == 1))
  TPRvals[iter] <- TP/(TP+FN)
  FPRvals[iter] <- FP/(FP+TN)
  iter = iter +1
}
plot(FPRvals, TPRvals,main = "Receiver Operator Characteristic",  type = "l",
     xlab = "FPR", ylab = "TPR")
```



ii.

```
AUC <- function(x,y){
  d <- data.frame(x,y)
  d <- d[order(d$x),]
  n <- nrow(d)
  sum(diff(d$x) * (d$y[-n]+d$y[-1]))/2
}

area <- AUC(FPRvals, TPRvals)

plot(FPRvals, TPRvals,main = "Receiver Operator Characteristic",  type = "l",
     xlab = "FPR", ylab = "TPR", lwd = 3)
polygon(x = c(1, FPRvals, 0), y = c(0, TPRvals, 0), border = NA, col=rgb(0.5, 0.5, 0.5, 0.5))


text(0.6, 0.4, paste("AUC =", round(area, 4)), col = "blue")
abline(0,1,lty=5)
```


Receiver Operator Characteristic
TPR
FPR
0.0
0.2
0.4
0.6
0.8
1.0
AUC = 0.9825

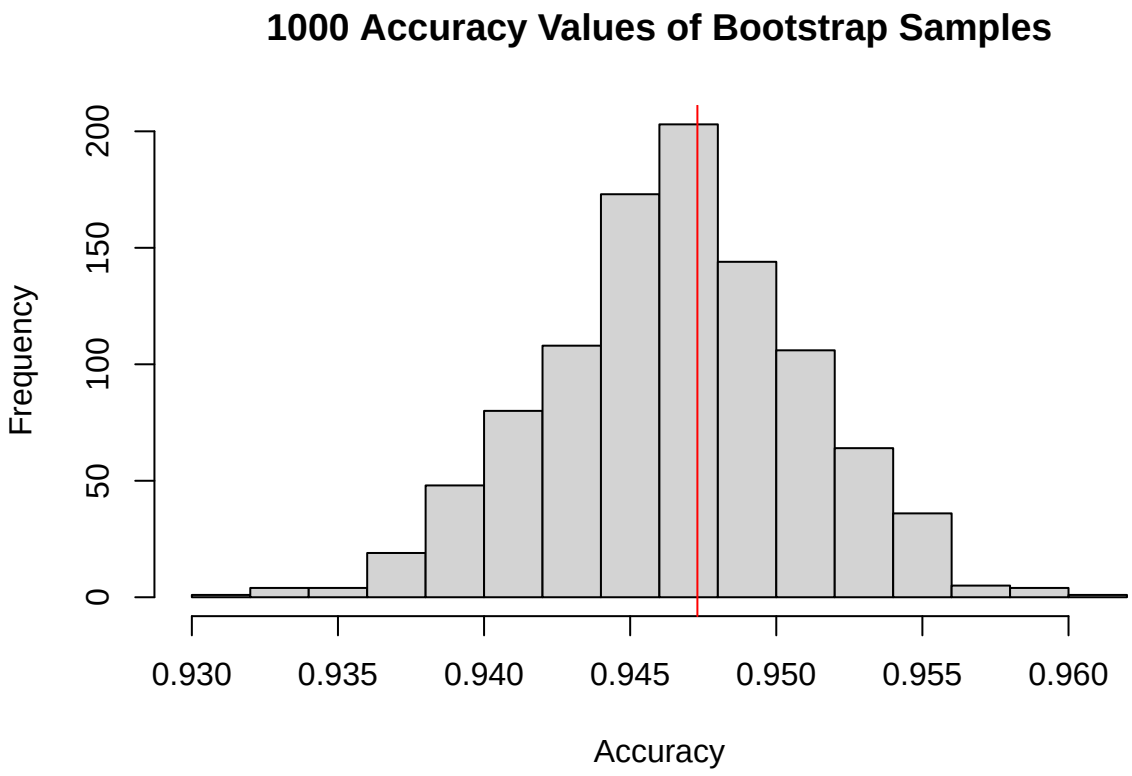
QUESTION 3: Bootstrap Confidence Intervals

(a)

```
B = 1000
n <- nrow(votes)
set.seed(341)
Sstar <- sapply(1:B, FUN = function(b) {
    sample(1:n, n, replace = TRUE)
})
```

(b) i.

```
acc_vals <- rep(0, B)

for (b in 1:B){
  index <- Sstar[,b]
  ys <- y[index]
  xs <- x[index]

  psiFns <- createLogisticPsiFns(xs,ys)
  alpha0 <- log(mean(y)/(1-mean(y)))
  beta0 <- 0
  outputnew <- NewtonRaphson(theta = c(alpha0, beta0), psiFn = psiFns$psiFn,
                             psiPrimeFn = psiFns$psiPrimeFn)
  phvals <- p_hat(xs, outputnew$theta)
  c <- 0.5
  yh <- (phvals >= c)*1

  confmatrix <- table(ys, yh)
  TN <- confmatrix[1, 1]
  TP <- confmatrix[2, 2]
  FP <- confmatrix[1, 2]
  FN <- confmatrix[2, 1]
  acc_vals[b] <- (TN + TP)/(TN + FP + FN + TP)
}

hist(acc_vals, main = "1000 Accuracy Values of Bootstrap Samples", xlab = "Accuracy")
abline(v=accuracy, col = "red")
```



ii.

```
accuracy + qnorm(0.975) * c(-1, 1) * sd(acc_vals)
```

```
## [1] 0.9386707 0.9559308
```

The 95% confidence interval for the true population accuracy using the naive normal theory approach is [0.9386707, 0.9559308].

iii.

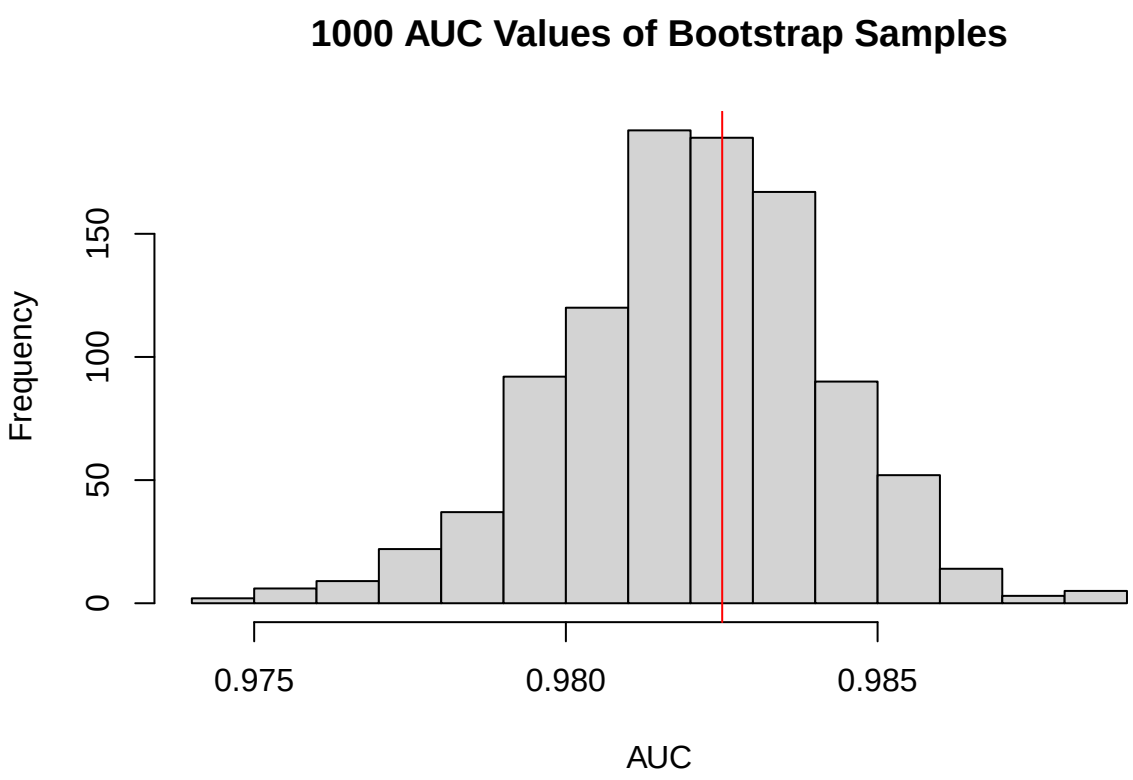
```
c(quantile(acc_vals, 0.025), quantile(acc_vals, 0.975))
```

```
##      2.5%     97.5%
## 0.9379820 0.9550129
```

The 95% confidence interval for the true population accuracy using the percentile method is [0.9379820, 0.9550129].

(c) i.

```
auc_vals <- rep(0, B)

for (b in 1:B){
  index <- Sstar[,b]
  ys <- y[index]
  xs <- x[index]

  psiFns <- createLogisticPsiFns(xs,ys)
  alpha0 <- log(mean(y)/(1-mean(y)))
  beta0 <- 0
  outputnew <- NewtonRaphson(theta = c(alpha0, beta0), psiFn = psiFns$psiFn,
                             psiPrimeFn = psiFns$psiPrimeFn)
  phvals <- p_hat(xs, outputnew$theta)

  c <- seq(from=0, to=1, length.out=500)
  TPRnew <- rep(0, length(c))
  FPRnew <- rep(0, length(c))

  iterations = 1
  for (inew in c){
    yh <- (phvals >= inew)*1
    TN <- sum((yh == 0) & (ys == 0))
    TP <- sum((yh == 1) & (ys == 1))
    FP <- sum((yh == 1) & (ys == 0))
    FN <- sum((yh == 0) & (ys == 1))

    TPRnew[iterations] <- TP/(TP+FN)
    FPRnew[iterations] <- FP/(FP+TN)
    iterations = iterations + 1
  }
  auc_vals[b] <- AUC(FPRnew, TPRnew)
}

hist(auc_vals, main = "1000 AUC Values of Bootstrap Samples", xlab = "AUC")
abline(v=area, col = "red")
```



ii.

```
area + qnorm(0.975) * c(-1, 1) * sd(auc_vals)
```

```
## [1] 0.9783040 0.9867108
```

The 95% confidence interval for the true population AUC using the naive normal theory approach is [0.9783040, 0.9867108].

iii.

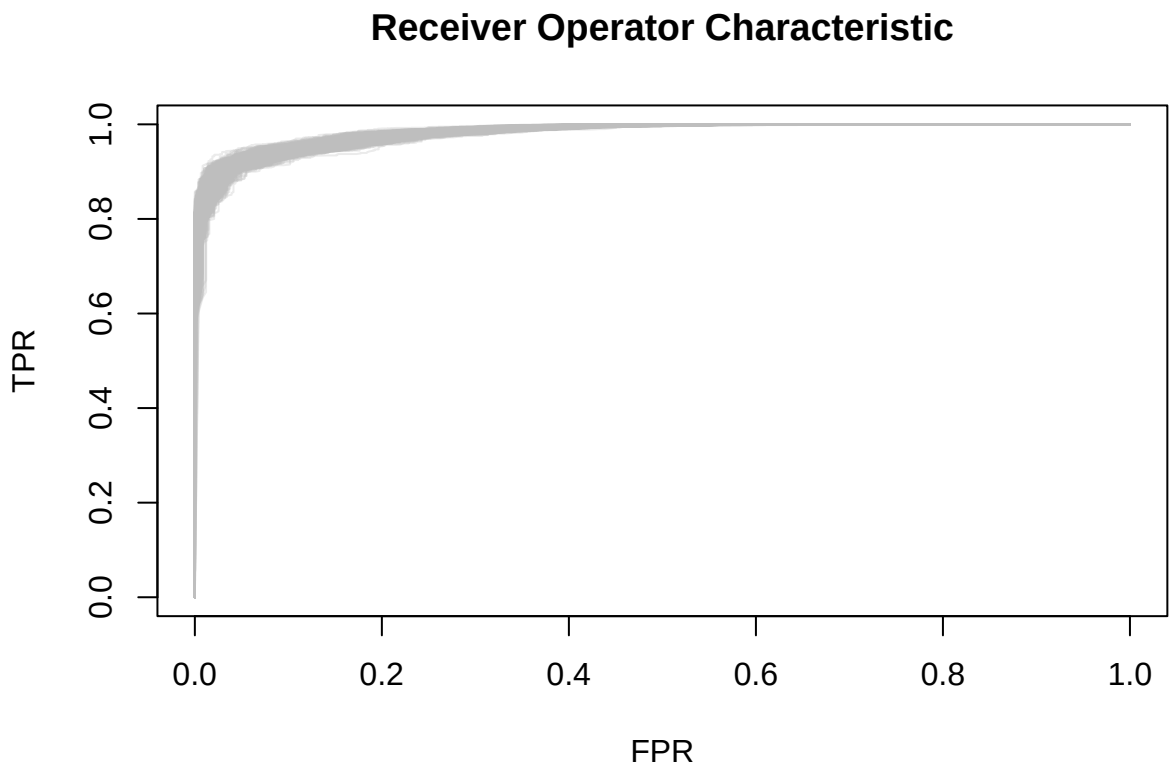
```
c(quantile(auc_vals, 0.025), quantile(auc_vals, 0.975))
```

```
##      2.5%     97.5%
## 0.9774424 0.9858739
```

The 95% confidence interval for the true population AUC using the percentile method is [0.9774424, 0.9858739].

(d) i.

```
plot(NA, main="Receiver Operator Characteristic", xlab = "FPR", ylab = "TPR",
     xlim = c(0,1), ylim = c(0,1))
for (b in 1:B){
  index <- Sstar[,b]
  ys <- y[index]
  xs <- x[index]

  psiFns <- createLogisticPsiFns(xs,ys)
  alpha0 <- log(mean(y)/(1-mean(y)))
  beta0 <- 0
  outputnew <- NewtonRaphson(theta = c(alpha0, beta0), psiFn = psiFns$psiFn,
                             psiPrimeFn = psiFns$psiPrimeFn)
  phvals <- p_hat(xs, outputnew$theta)

  c <- seq(from=0, to=1, length.out=500)
  TPRnew <- rep(0, length(c))
  FPRnew <- rep(0, length(c))

  iterations = 1
  for (inew in c){
    yh <- (phvals >= inew)*1
    TN <- sum((yh == 0) & (ys == 0))
    TP <- sum((yh == 1) & (ys == 1))
    FP <- sum((yh == 1) & (ys == 0))
    FN <- sum((yh == 0) & (ys == 1))

    TPRnew[iterations] <- TP/(TP+FN)
    FPRnew[iterations] <- FP/(FP+TN)
    iterations = iterations + 1
  }
  lines(FPRnew, TPRnew, col = adjustcolor("grey", 0.3))
}
```



(e)

From question 2, we know the point estimate of accuracy is 0.9473008 and that of AUC is 0.9825. The value of AUC is 0.9825, which indicates that there is around 98% situations that based on the data in 2012 (i.e. the proportion of votes in the county received by Obama in 2012), it is predictable who they will vote for in 2016.
The 95% confidence interval for the true population accuracy using the naive normal theory approach is [0.9386707, 0.9559308]. The 95% confidence interval for the true population accuracy using the percentile method is [0.9379820, 0.9550129]. The 95% confidence interval for the true population AUC using the naive normal theory approach is [0.9783040, 0.9867108]. The 95% confidence interval for the true population AUC using the percentile method is [0.9774424, 0.9858739]. These four confidence intervals also have high lower bound and upper bound. Thus, the logistic regression model is a good model and it is reasonable that we could predict the the number of votes in the country received by each candidate based on the election result in 2012. Thus, the ability of logistic regression to predict the winner of a county, given the voting behaviour in that county in the previous election, is good. The past voting behaviour does appear to be predictive of future voting behaviour.