Assignment1

Regina Crespo Lopez Oliver (20000322-T884) & Malin Mueller (20011115-T460)

2024-09-27

R Markdown

```
# Byt ÅÅMMDD mot ditt födelsedatum
set.seed(011115) # - Malin
# set.seed(000322) # - Regina
```

Task 1

```
load("proj_data.Rdata")
modell <- glm(Resultat ~ Alder + Kon + Utbildare,
              data = data_individ,
              family = "binomial")
summary(modell)
```

```
##
## Call:
## glm(formula = Resultat ~ Alder + Kon + Utbildare, family = "binomial",
##     data = data_individ)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           0.151971   0.249661   0.609 0.542716
## Alder                -0.031394   0.008677  -3.618 0.000297 ***
## KonMan                0.090185   0.136394   0.661 0.508476
## UtbildareTrafikskola  0.916541   0.157659   5.813 6.12e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1353  on 999  degrees of freedom
## Residual deviance: 1297  on 996  degrees of freedom
## AIC: 1305
##
## Number of Fisher Scoring iterations: 4
```

```r
y <- matrix(data_individ$Resultat, ncol = 1)
X <- model.matrix(Resultat ~ Alder + Kon + Utbildare,
                  data = data_individ)
head(X)
```

```
##      (Intercept) Alder KonMan UtbildareTrafikskola
## 3316           1    19      0                    1
## 9863           1    30      1                    1
## 6186           1    21      1                    0
## 1154           1    18      0                    1
## 3265           1    19      1                    1
## 9757           1    30      1                    0
```

```r
head(data_individ[, -1])
```

```
##         Kon Alder   Utbildare
## 3316 Kvinna    19 Trafikskola
## 9863    Man    30 Trafikskola
## 6186    Man    21   Privatist
## 1154 Kvinna    18 Trafikskola
## 3265    Man    19 Trafikskola
## 9757    Man    30   Privatist
```

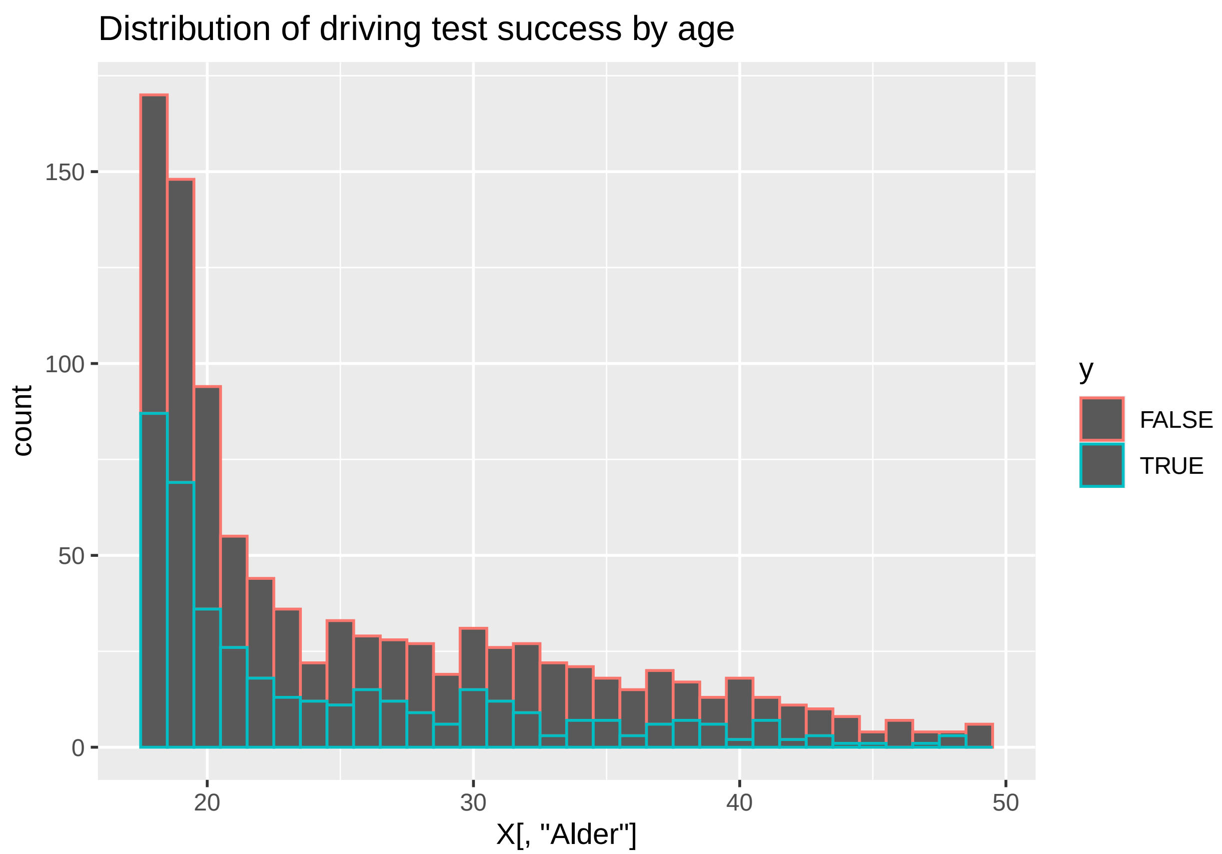
Extra : Data visualization

```r
library(ggplot2)

# x_true <- X[y == TRUE,]

ggplot(X, aes(X[,"Alder"], colour = y)) +

  geom_histogram(bins = 32)  +
  ggtitle("Distribution of driving test success by age")
```



From the graph we can see that most applicants to the test are younger. There is always a high chance of failure in each age group.

Task 2

```
p_var <- function(theta, X){
  # function to compute probabiloty recieving theta and X
  p_res <- 1 / (1 + exp(-X%*%theta))
  return(p_res)
}

L <- function(theta, y, X){
  ## likelihood
  p_var <- p_var(theta = theta, X = X)
  likelihood <- prod(p_var^y * (1 - p_var)^(1 - y))
  return(likelihood)
}

l <- function(theta, y, X){
  ## Same thing as before, but log likelihood
  ## log likelihood
  p_var <- p_var(theta = theta, X = X)
  log_likelihood <- sum(y * log(p_var) + (1 - y) * log(1 - p_var))
  # log_likelihood <- dbninom()
  return(log_likelihood)
```

```r
}

S <- function(theta, y, X){
  ## Score function
  p_var <- p_var(theta = theta, X = X)
  score <- t(X) %*% (y-p_var)
  return(score)
}

v <- function(theta, X){
  ## Vi
  p <- p_var(theta = theta, X = X)
  v_res <- p * (1-p)
  return(v_res)
}

I <- function(theta, y, X){
  ## fisher information
  v_var = v(theta,X)
  D <- diag(as.vector(v_var))
  fisher <- t(X) %*% D %*% X
  return(fisher)
  }

NR <- function(theta0, niter, y, X){
  # function that applies Newton-Raphson?s algorithm in order to compute the
  # ML-estimates in a logistic regression model, in a certain number of n
  # iterations.

  #Note: this might break if the matrix dim from x changes
    theta <- matrix(theta0, nrow = 4, ncol = 1)

  for (i in 1:niter){
    score <- S(theta, y, X)
    log_likelihood <- L(theta, y, X)
    #theta <- theta + (log_likelihood/score)
    theta <- theta + solve(I(theta, y, X)) %*% score # its a plus and not a
    # minus because of how we obtain the derivative of the score function
  }

  return(theta)
}
```

```r
theta0 <- c(0, 0, 0, 0)
NR_estimate <- NR(theta0, 5, y, X)

print(NR_estimate)
```

```
##                              [,1]
## (Intercept)            0.15197099
## Alder                 -0.03139396
## KonMan                 0.09018524
## UtbildareTrafikskola   0.91654080
```

```r
# Function to compute the number of iterations for two-digit accuracy
find_niter_NR <- function(theta0, y, X, tol = 1e-2) {

  glm_coef <- coef(modell)  # Extract coefficients from glm

  # Initial guess for niter and starting point for NR
  niter <- 1
  max_iter <- 10  # Set a maximum number of iterations,
  # prevents from looping forever

  # Loop until the difference between NR and glm is within 2 digits (tol = 1e-2)
  while (niter <= max_iter) {
    theta_NR <- NR(theta0, niter, y, X)

    # check the match
    if (all(abs(round(theta_NR, 2) - round(glm_coef, 2)) < tol)) {
      return(niter)  # Return the number of iterations needed
    }

    # Increase iteration count
    niter <- niter + 1
  }

  # If max_iter is reached and no convergence, return a message
  return(paste("Did not converge within", max_iter, "iterations"))
}


# Starting value for theta0 (initial guess for NR)
theta0 <- c(0, 0, 0, 0)

# Find the number of iterations needed for NR to match glm's estimates within two digits
niter_needed <- find_niter_NR(theta0, y, X)
print(niter_needed)
```

```
## [1] 2
```

The number of iterations needed is 2.

Task 3

Approximate standard error of our estimate:

```r
I_mat = I(NR_estimate, y, X) # first obtain the fisher information
inv_I = diag(solve(I_mat)) #diagonal of the elements of the inverse fisher info.
NR_error = sqrt(inv_I) # then we obtain the square root
NR_error
```

```
##          (Intercept)                Alder               KonMan
##          0.249660636          0.008677277          0.136393699
## UtbildareTrafikskola
##          0.157658653
```

Standard error given by R:

```r
summary(modell)$coefficients[,"Std. Error"]
```

```
##          (Intercept)                Alder               KonMan
##          0.249660565          0.008677273          0.136393673
## UtbildareTrafikskola
##          0.157658638
```

The standard error given by R is very close to that of our approximation, indicating that R may be using the same method as us.

The following computation is made to shown the difference between our approximation and that made by R.

```r
abs(NR_error - summary(modell)$coefficients[,"Std. Error"])
```

```
##          (Intercept)                Alder               KonMan
##         7.022601e-08         4.290470e-09         2.655224e-08
## UtbildareTrafikskola
##         1.416865e-08
```

Task 4

```r
parametric_bootstrap <- function(X, y, n_bootstrap = 1000) {
  # the parametric bootstrap function takes new samples of x from the original modell
  # to create new yi´s (resultat) and draw thetahat estiamtes from that
  # finally, after n bootstrap runs, we take the standard deviations for each theta

  n <- length(y)
  theta_hat <- coef(modell)  # estimates from glm
  prob_hat <- p_var(theta_hat, X)  # Predicted probabilities

  # MAtrix full of NAs to be replaced by bootstrap estimates
  bootstrap_estimates <- matrix(NA, n_bootstrap, length(theta_hat))

  # Perform bootstrap sampling
  for (i in 1:n_bootstrap) {
    # Generate new y values based on the fitted probabilities prob_hat
    # from Bernoulli (binomial with n=1)
    y_boot <- rbinom(n, 1, prob = prob_hat)

    # Fit glm on bootstrapped sample
    modell_boot <- glm(y_boot ~ Alder + Kon + Utbildare,
                data = data_individ,
                family = "binomial")

    # Store the estimated coefficients (here theta) in each row
    bootstrap_estimates[i, ] <- coef(modell_boot)
  }
```

```r
  # Calculate standard errors as the standard deviation of bootstrap estimates
  # 2 means you take the standard deviation of each column
  se_boot <- apply(bootstrap_estimates, 2, sd)
  return(se_boot)
}

se_bootstrap <- parametric_bootstrap(X, y, n_bootstrap = 1000)
print(se_bootstrap)
```

```
## [1] 0.253930333 0.008826321 0.129857635 0.156386189
```

The values from the bootstrap are close, but slightly different to the standard error of R.

```r
# construct 95% CI with bootstrap

bootstrap_ci <- function(X, y, age, sex, education, n_bootstrap = 1000) {
  # we use the normal bootstrapped sample like in the above function,
  # and calculate the probability (1 / (1 + exp(-X%*%theta_hat)))
  # for each run of the bootstrap
  n <- length(y)
  theta_hat <- coef(modell)  # estimates from glm
  prob_hat <-  p_var(theta_hat, X)  # Fitted probabilities

  # Construct the new Xi for the person with my age
  new_data <- c(1, age, sex, education)

  # Storage vector for bootstrap probabilities
  prob_bootstrap <- numeric(n_bootstrap)

  # Perform bootstrap sampling
  for (i in 1:n_bootstrap) {

    y_boot <- rbinom(n, 1, prob = prob_hat)

    modell_boot <- glm(y_boot ~ Alder + Kon + Utbildare,
              data = data_individ,
              family = "binomial")

    # Predict the probability for the new data points
    prob_bootstrap[i] <- p_var(coef(modell_boot), new_data)
  }

  # Calculate 95% confidence interval from the bootstrap probabilities
  ci_lower <- quantile(prob_bootstrap, 0.025)
  ci_upper <- quantile(prob_bootstrap, 0.975)

  return(c(ci_lower, ci_upper))
}

ci <- bootstrap_ci(X, y, age = 22, sex = 0, education = 1, n_bootstrap = 1000)
print(ci)
```

```
##      2.5%     97.5%
```

```
## 0.5265742 0.6608923
```

95% CI is (0.526, 0.656)

The probability of Malin (age 22) of passing the exam is between .53 and .66.

```r
ci <- bootstrap_ci(X, y, age = 24, sex = 0, education = 1, n_bootstrap = 1000)
print(ci)
```

```
##      2.5%     97.5%
## 0.5096248 0.6461037
```

For Regina's age (24) the C.I of passing is between 0.51 and 0.64.