practical_exercise_1_solution.Rmd

Lau Møller Andersen

6/9/2021

Solutions for practical exercise 1

Exercise 1

1. extract $\hat{\beta}$, Y, $\hat{Y}$, X and ϵ from **model** (hint: have a look at the function **model.matrix**)
    i. create a plot that illustrates Y and $\hat{Y}$ (if you are feeling ambitious, also include ϵ (hint:

```
data(mtcars)
model <- lm(mpg ~ wt, data=mtcars)

## extracting the parameters from the model object
print(beta.hat <- model$coefficients)
```

```
## (Intercept)          wt
##   37.285126   -5.344472
```

```
print(Y <- model$model$mpg)
```

```
##  [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2 10.4
## [16] 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7
## [31] 15.0 21.4
```

```
print(Y.hat <- model$fitted.values)
```

```
##           Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
##           23.282611           21.919770           24.885952           20.102650
##   Hornet Sportabout             Valiant          Duster 360           Merc 240D
##           18.900144           18.793255           18.205363           20.236262
##            Merc 230            Merc 280           Merc 280C          Merc 450SE
##           20.450041           18.900144           18.900144           15.533127
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
##           17.350247           17.083024            9.226650            8.296712
##   Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla
##            8.718926           25.527289           28.653805           27.478021
##       Toyota Corona    Dodge Challenger         AMC Javelin          Camaro Z28
##           24.111004           18.472586           18.926866           16.762355
##    Pontiac Firebird           Fiat X1-9       Porsche 914-2        Lotus Europa
##           16.735633           26.943574           25.847957           29.198941
##      Ford Pantera L        Ferrari Dino       Maserati Bora          Volvo 142E
##           20.343151           22.480940           18.205363           22.427495
```

```
print(X <- model.matrix(model))
```

```
##                     (Intercept)    wt
## Mazda RX4                     1 2.620
```

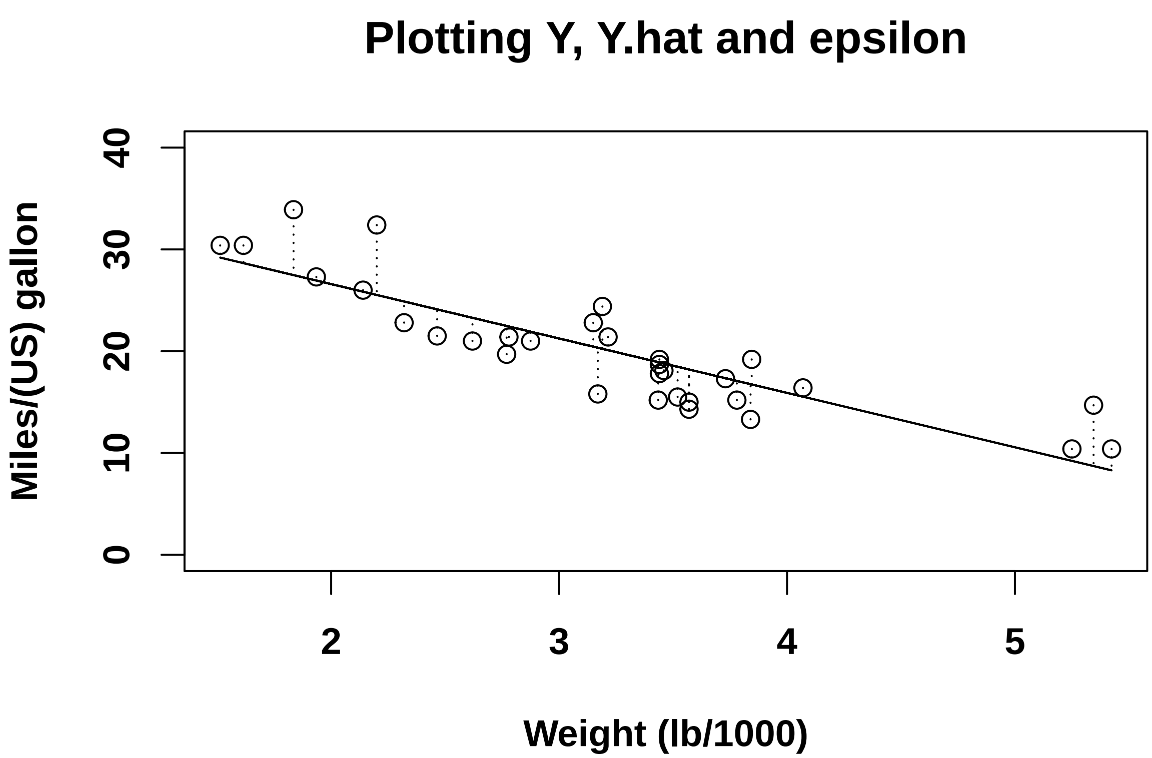
```
## Mazda RX4 Wag              1 2.875
## Datsun 710                 1 2.320
## Hornet 4 Drive             1 3.215
## Hornet Sportabout          1 3.440
## Valiant                    1 3.460
## Duster 360                 1 3.570
## Merc 240D                  1 3.190
## Merc 230                   1 3.150
## Merc 280                   1 3.440
## Merc 280C                  1 3.440
## Merc 450SE                 1 4.070
## Merc 450SL                 1 3.730
## Merc 450SLC                1 3.780
## Cadillac Fleetwood         1 5.250
## Lincoln Continental        1 5.424
## Chrysler Imperial          1 5.345
## Fiat 128                   1 2.200
## Honda Civic                1 1.615
## Toyota Corolla             1 1.835
## Toyota Corona              1 2.465
## Dodge Challenger           1 3.520
## AMC Javelin                1 3.435
## Camaro Z28                 1 3.840
## Pontiac Firebird           1 3.845
## Fiat X1-9                  1 1.935
## Porsche 914-2              1 2.140
## Lotus Europa               1 1.513
## Ford Pantera L             1 3.170
## Ferrari Dino               1 2.770
## Maserati Bora              1 3.570
## Volvo 142E                 1 2.780
## attr(,"assign")
## [1] 0 1
```

```r
print(epsilon <- model$residuals)
```

```
##           Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
##          -2.2826106          -0.9197704          -2.0859521           1.2973499
##   Hornet Sportabout             Valiant          Duster 360           Merc 240D
##          -0.2001440          -0.6932545          -3.9053627           4.1637381
##            Merc 230            Merc 280           Merc 280C          Merc 450SE
##           2.3499593           0.2998560          -1.1001440           0.8668731
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
##          -0.0502472          -1.8830236           1.1733496           2.1032876
##   Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla
##           5.9810744           6.8727113           1.7461954           6.4219792
##       Toyota Corona    Dodge Challenger         AMC Javelin          Camaro Z28
##          -2.6110037          -2.9725862          -3.7268663          -3.4623553
##    Pontiac Firebird           Fiat X1-9       Porsche 914-2        Lotus Europa
##           2.4643670           0.3564263           0.1520430           1.2010593
##      Ford Pantera L        Ferrari Dino       Maserati Bora          Volvo 142E
##          -4.5431513          -2.7809399          -3.2053627          -1.0274952
```

```r
## plotting Y and Y.hat together with the errors
par(font.lab=2, font.axis=2, cex=1.2)
```

```r
plot(mtcars$wt, Y, ylim=c(0, 40), xlab='Weight (lb/1000)',
     ylab='Miles/(US) gallon', main='Plotting Y, Y.hat and epsilon')
lines(mtcars$wt, Y.hat)
n.obs <- dim(mtcars)[1]
for(index in 1:n.obs)
{
    x <- mtcars$wt[index]
    y0 <- Y[index]
    y1 <- Y.hat[index]

    lines(c(x, x), c(y0, y1), lty=3)
}
```



2. estimate β for a quadratic model ($y = \beta_2 x^2 + \beta_1 x + \beta_0$) using ordinary least squares *without* using **lm**; $\hat{\beta} = (X^T X)^{-1} X^T Y$ (hint: add a third column to X from step 1)

```r
X.quad <- cbind(X, X[, 2]^2)
beta.hat.quad <- solve(t(X.quad) %*% X.quad) %*% t(X.quad) %*% Y
model.quad <- lm(mpg ~ wt + I(wt^2) + 1, data=mtcars)
print(beta.hat.quad)
```

```
##                  [,1]
## (Intercept)  49.930811
## wt          -13.380337
##               1.171087
```

3. compare your acquired $\hat{\beta}$ with the output of the corresponding quadratic model created using **lm**

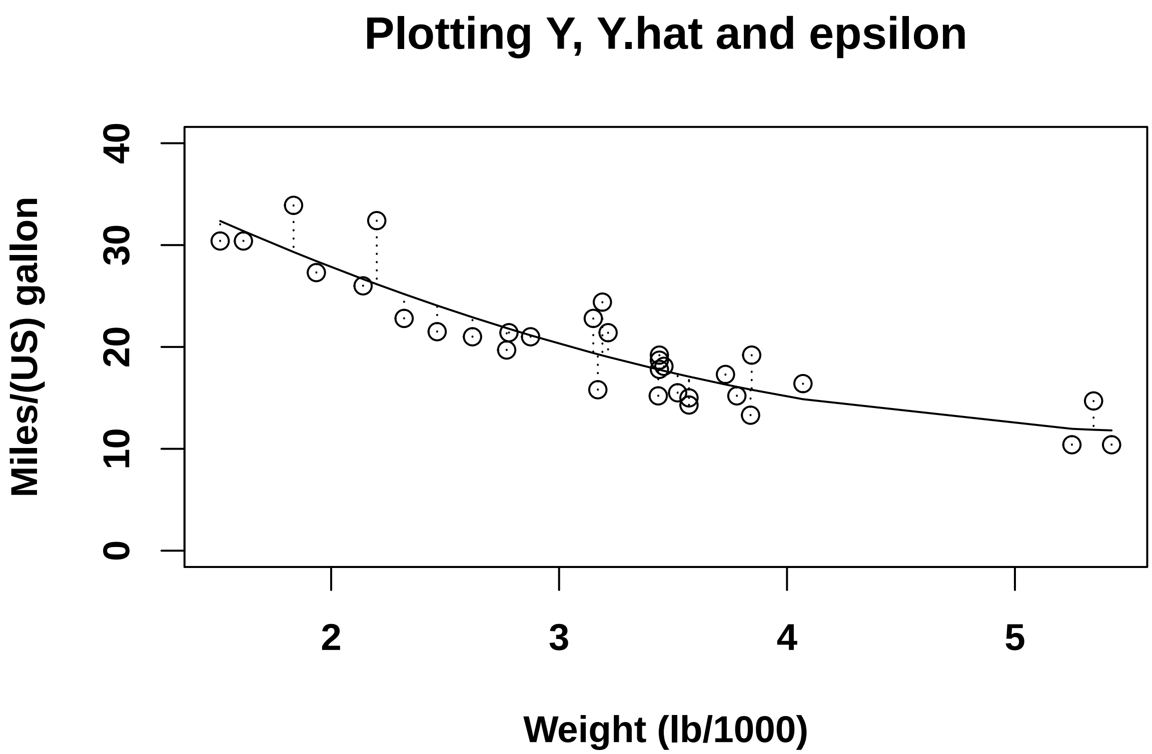
(hint: use the function **I**, see details under help and the sub-section formula operators here: https://www.datacamp.com/community/tutorials/r-formula-tutorial)

i. create a plot that illustrates Y and $\hat{Y}$ (if you are feeling ambitious, also include ϵ (hint: you can use the function **arrows**))

```
print(model.quad)
```

```
##
## Call:
## lm(formula = mpg ~ wt + I(wt^2) + 1, data = mtcars)
##
## Coefficients:
## (Intercept)           wt      I(wt^2)
##      49.931      -13.380        1.171
```

```
## plotting Y and Y.hat together with the errors
par(font.lab=2, font.axis=2, cex=1.2)
plot(mtcars$wt, Y, ylim=c(0, 40), xlab='Weight (lb/1000)',
     ylab='Miles/(US) gallon', main='Plotting Y, Y.hat and epsilon')
Y.hat <- model.quad$fitted.values
## we need to sort the values to use "lines"
sort.list <- sort(mtcars$wt, index.return=TRUE)
lines(sort.list$x, Y.hat[sort.list$ix])

n.obs <- dim(mtcars)[1]
for(index in 1:n.obs)
{
    x <- mtcars$wt[index]
    y0 <- Y[index]
    y1 <- Y.hat[index]

    lines(c(x, x), c(y0, y1), lty=3)
}
```



Exercise 2

1. which seems better?
   It seems the fit is better for the quadratic fit

2. calculate the sum of squared errors, (show the calculation based on ϵ). Which fit has the lower sum?

```
epsilon.linear <- model$residuals
epsilon.quad <- model.quad$residuals

print(SS.linear <- sum(epsilon.linear^2))
```

```
## [1] 278.3219
```

```
print(SS.quad <- sum(epsilon.quad^2))
```

```
## [1] 203.7454
```

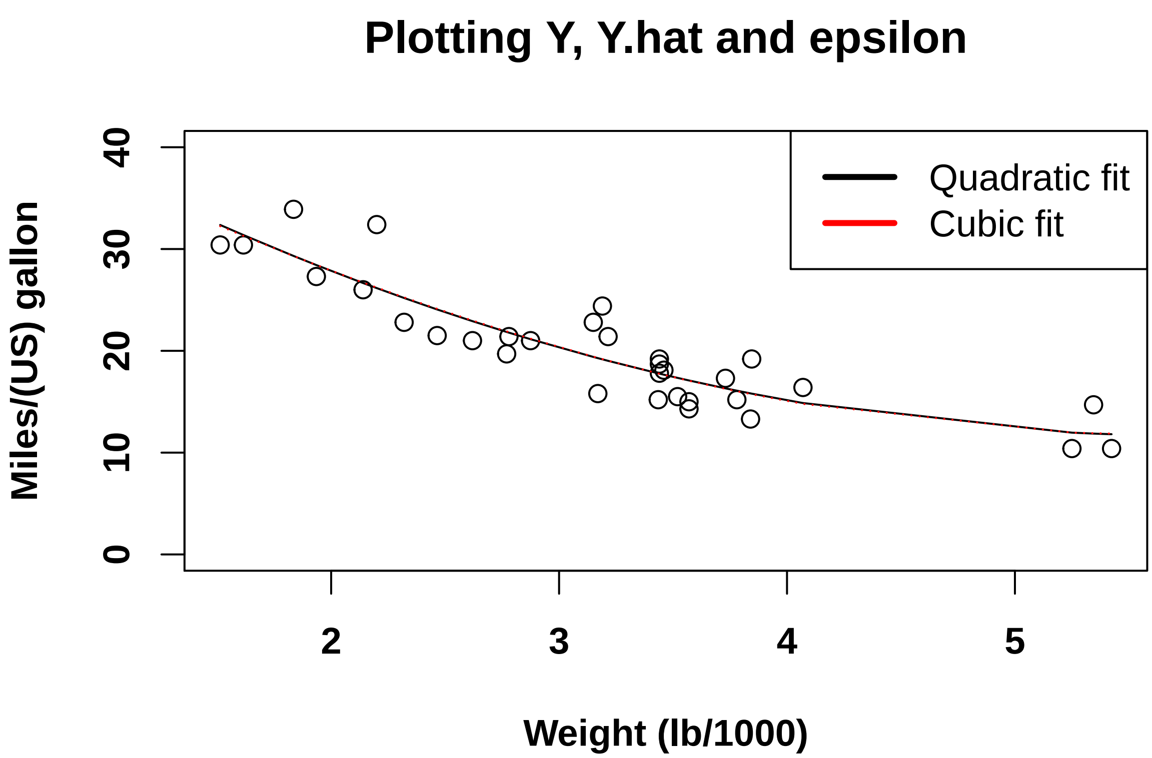
The sum of squared errors is smaller for the quadratic fit

3. now make a cubic fit ($y = \beta_3 x^3 + \beta_2 x^2 + \beta_1 x + \beta_0$) and compare it to the quadratic fit
   i. create a plot that illustrates Y and $\hat{Y}$ for both the cubic and the quadratic fits (plot them in the same plot)

   ii. compare the sum of squared errors

   iii. what's the estimated value of the "cubic" (β_3) parameter? Comment on this!

```r
model.cub <- lm(mpg ~ I(wt^3) + I(wt^2) + wt + 1, data=mtcars)
## plotting Y and Y.hat together with the errors
par(font.lab=2, font.axis=2, cex=1.2)
plot(mtcars$wt, Y, ylim=c(0, 40), xlab='Weight (lb/1000)',
     ylab='Miles/(US) gallon', main='Plotting Y, Y.hat and epsilon')
Y.hat.quad <- model.quad$fitted.values
Y.hat.cub <- model.cub$fitted.values
## we need to sort the values to use "lines"
sort.list. <- sort(mtcars$wt, index.return=TRUE)

lines(sort.list$x, Y.hat.quad[sort.list$ix])
lines(sort.list$x, Y.hat.cub[sort.list$ix], lty=3, col='red')

legend('topright', c('Quadratic fit', 'Cubic fit'), col=c('black', 'red'),
       lwd=3)
```




```r
epsilon.cub <- model.cub$residuals
print(SS.quad <- sum(epsilon.quad^2))
```

```
## [1] 203.7454
```

```r
print(SS.cub <- sum(epsilon.cub^2))
```

```
## [1] 203.6699
```

```r
print(beta.hat.3 <- model.cub$coefficients[2])
```

```
##    I(wt^3)
```

```
## 0.04593618
```

SS.cub is smaller than SS.quad but only very little so, indicating that the cubic part doesn't contribute a lot, as is also indicated by the small size of beta.hat.3. Note, that this isn't a reliable measure if the other beta.hats also change a lot

4. bonus question: which summary statistic is the fitted value (*Intercept* or β_0 in $y = \beta_0$) below identical to?

```
model.intercept <- lm(mpg ~ 1, data=mtcars)
print(mean(mtcars$mpg))
```

```
## [1] 20.09062
```

```
print(model.intercept$coefficients)
```

```
## (Intercept)
##    20.09062
```

Exercise 3

1. plot the fitted values for **logistic.model**:
    i. what is the relation between the **linear.predictors** and the **fitted_values** of the **logistic.model**

```
logit <-     function(x) log(x / (1 - x))
inv.logit <- function(x) exp(x) / (1 + exp(x))

logistic.model <- glm(am ~ wt + 1, data=mtcars, family='binomial')

print(logistic.model$fitted.values)
```

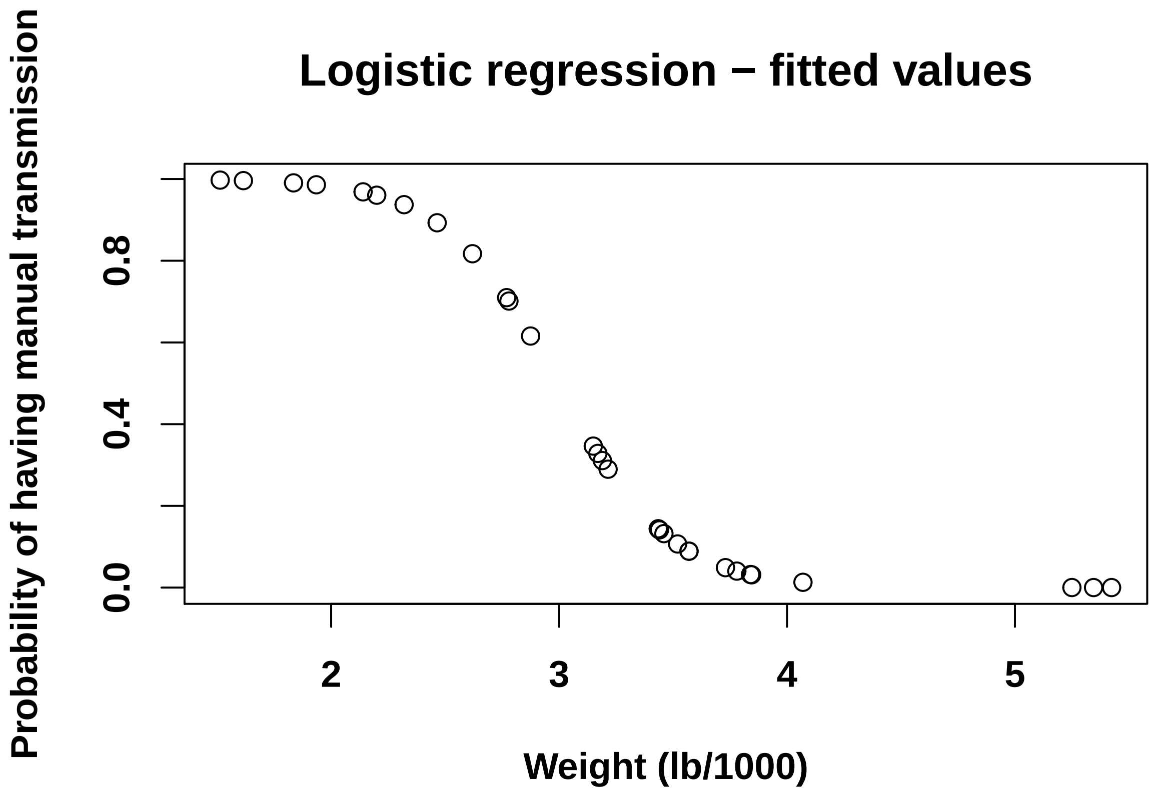
```
##           Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
##        8.172115e-01        6.157283e-01        9.373069e-01        2.897304e-01
##   Hornet Sportabout             Valiant          Duster 360           Merc 240D
##        1.415972e-01        1.320944e-01        8.905706e-02        3.108616e-01
##            Merc 230            Merc 280           Merc 280C          Merc 450SE
##        3.463470e-01        1.415972e-01        1.415972e-01        1.290453e-02
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
##        4.884439e-02        4.030126e-02        1.132870e-04        5.625028e-05
##   Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla
##        7.729920e-05        9.603663e-01        9.960953e-01        9.905887e-01
##       Toyota Corona    Dodge Challenger         AMC Javelin          Camaro Z28
##        8.929547e-01        1.067855e-01        1.440604e-01        3.193258e-02
##    Pontiac Firebird           Fiat X1-9       Porsche 914-2        Lotus Europa
##        3.131644e-02        9.859916e-01        9.686009e-01        9.974064e-01
##      Ford Pantera L        Ferrari Dino       Maserati Bora          Volvo 142E
##        3.283593e-01        7.097094e-01        8.905706e-02        7.013497e-01
```

```
print(inv.logit(logistic.model$linear.predictors))
```

```
##           Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
##        8.172115e-01        6.157283e-01        9.373069e-01        2.897304e-01
##   Hornet Sportabout             Valiant          Duster 360           Merc 240D
##        1.415972e-01        1.320944e-01        8.905706e-02        3.108616e-01
##            Merc 230            Merc 280           Merc 280C          Merc 450SE
##        3.463470e-01        1.415972e-01        1.415972e-01        1.290453e-02
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
##        4.884439e-02        4.030126e-02        1.132870e-04        5.625028e-05
```

```
##    Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla
##         7.729920e-05        9.603663e-01        9.960953e-01        9.905887e-01
##        Toyota Corona    Dodge Challenger         AMC Javelin          Camaro Z28
##         8.929547e-01        1.067855e-01        1.440604e-01        3.193258e-02
##     Pontiac Firebird           Fiat X1-9       Porsche 914-2        Lotus Europa
##         3.131644e-02        9.859916e-01        9.686009e-01        9.974064e-01
##       Ford Pantera L        Ferrari Dino       Maserati Bora          Volvo 142E
##         3.283593e-01        7.097094e-01        8.905706e-02        7.013497e-01
```

```
par(font.lab=2, font.axis=2, cex=1.2)
plot(mtcars$wt, logistic.model$fitted.values, xlab='Weight (lb/1000)',
     ylab='Probability of having manual transmission',
     main='Logistic regression - fitted values')
```

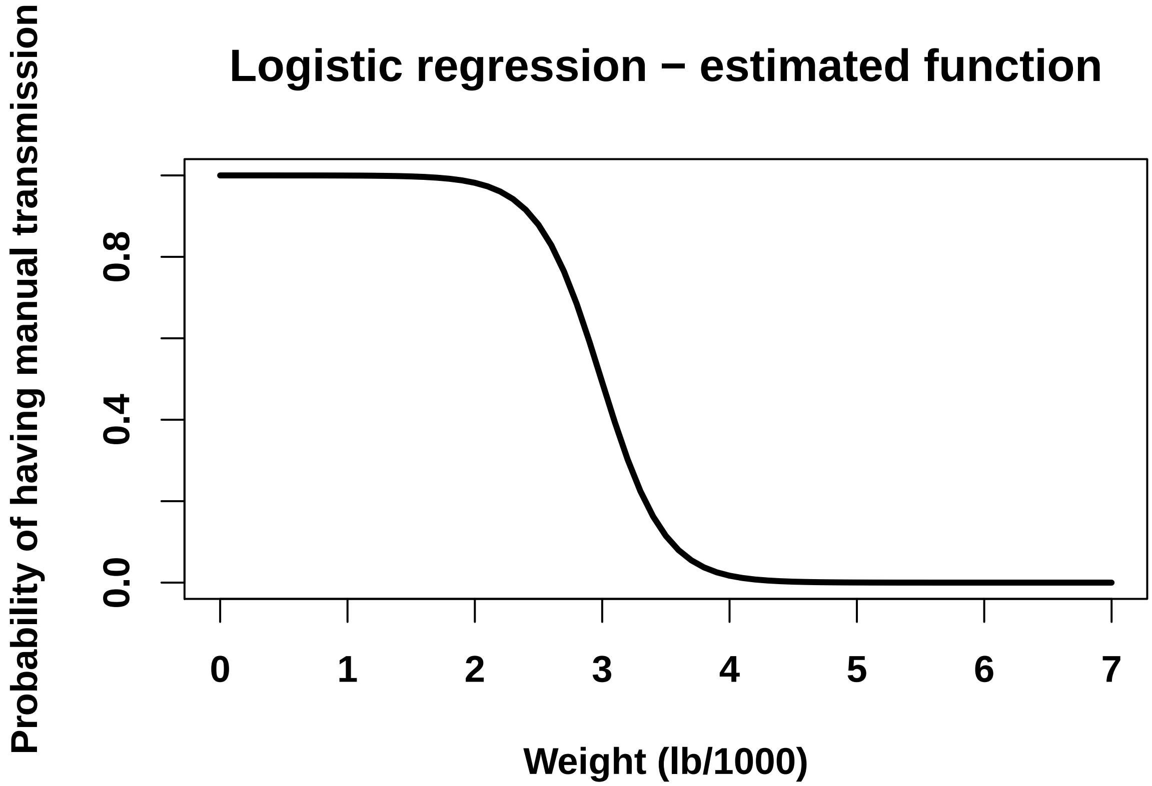


The fitted values are the linear predictors with the inv.logit applied

2. plot the logistic function, you've estimated based on your $\hat{\beta}$, (not just the fitted values). Use an *xlim* of (0, 7)
    i. what's the interpretation of the estimated $\hat{\beta}_0$ (the *Intercept*)
    ii. calculate the estimated probability that the Pontiac Firebird has automatic transmission, given its weight
    iii. bonus question - plot the logistic function and highlight all the cars where we guessed wrongly, if we used the following "quantizer" function:

$$transmission_{guess} = \begin{cases} 1(manual), & \text{if } PR(y=1) \geq 0.5 \\ 0(automatic), & \text{otherwise} \end{cases} \tag{1}$$

```r
# question i
wt <- seq(0, 7, 0.1)
p.am <- inv.logit(logistic.model$coefficients[1] +
    wt * logistic.model$coefficients[2])
par(font.lab=2, font.axis=2, cex=1.2)
plot(wt, p.am, type='l', lwd=3, xlab='Weight (lb/1000)',
     ylab='Probability of having manual transmission',
     main='Logistic regression - estimated function')
```




```r
print(inv.logit(logistic.model$coefficients[1])) ## the probability of a car with weight 0 (!) having ma
```

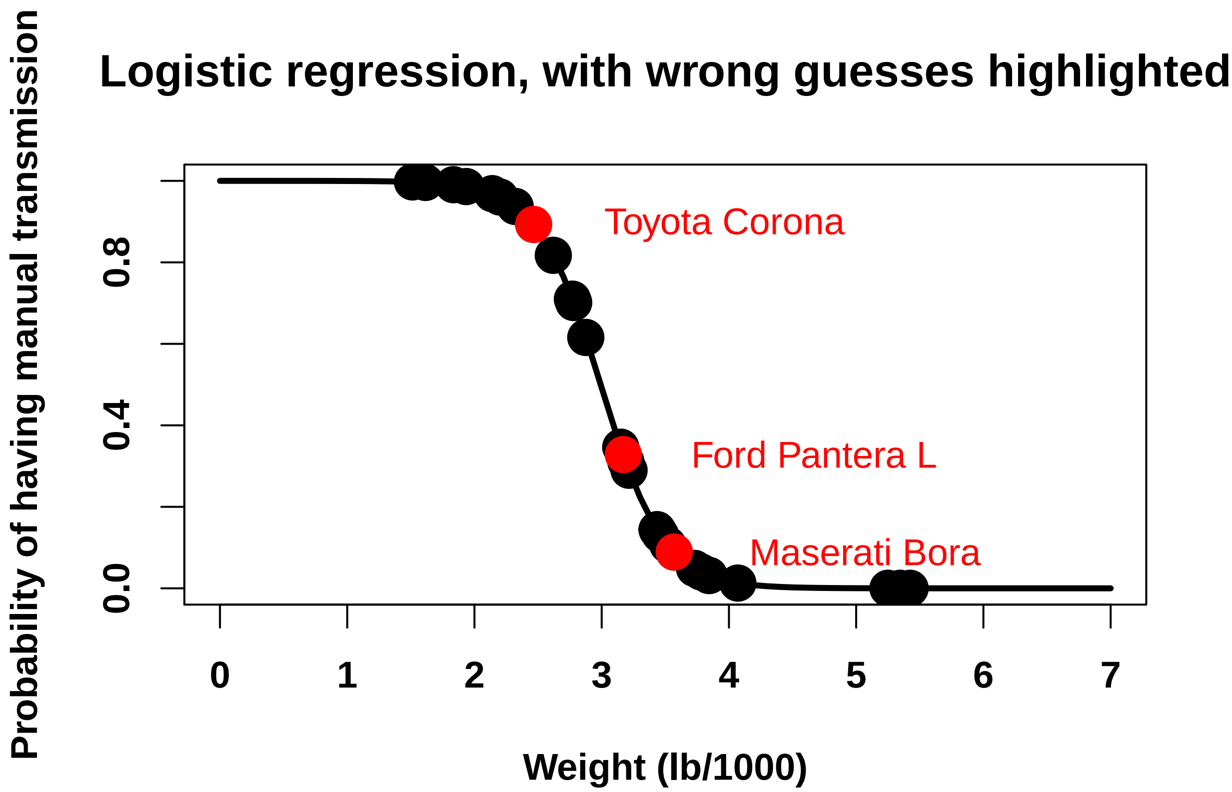
```
## (Intercept)
##   0.9999941
```

```r
# question ii
pf.index <- which(rownames(mtcars) == 'Pontiac Firebird')
print(p.pf <- inv.logit(logistic.model$coefficients[1] +
    mtcars$wt[pf.index] * logistic.model$coefficients[2]))
```

```
## (Intercept)
##  0.03131644
```

```r
# question iii


plot(wt, p.am, type='l', lwd=3, xlab='Weight (lb/1000)',
     ylab='Probability of having manual transmission',
```

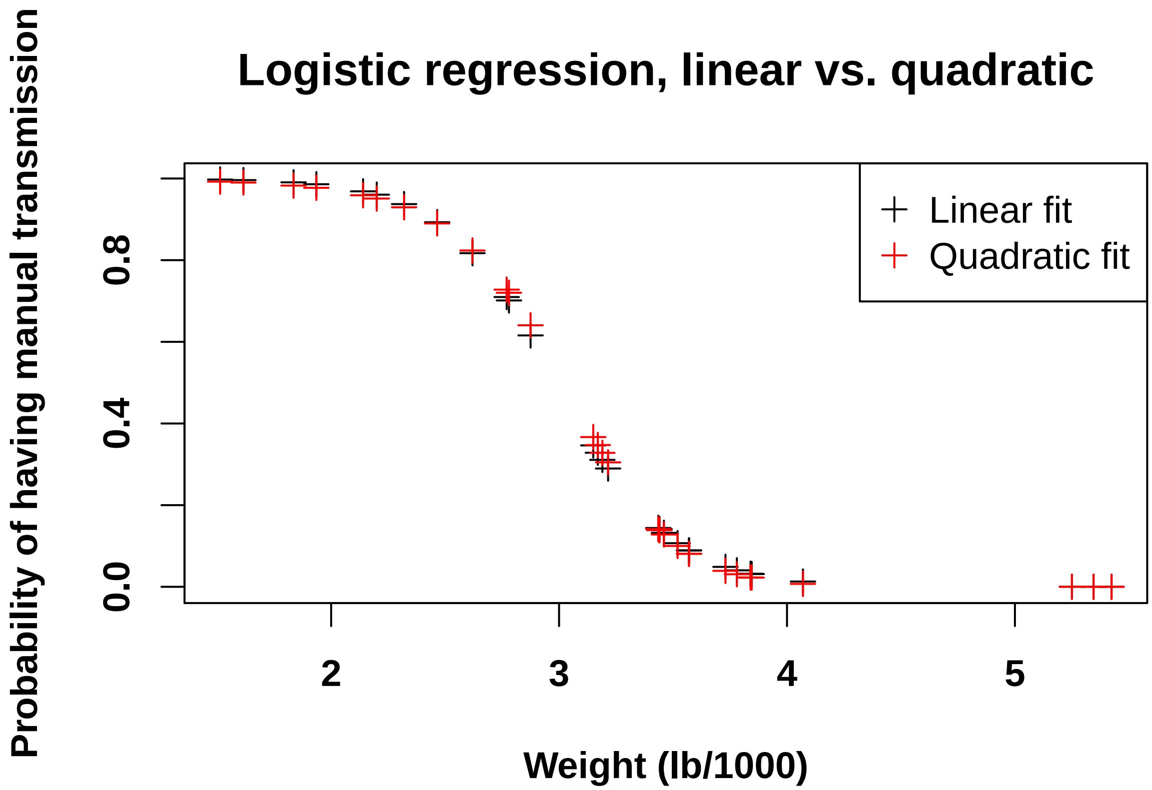
```
     main='Logistic regression, with wrong guesses highlighted')

mtcars$am.guess <- NA
n.obs <- dim(mtcars)[1]
for(index in 1:n.obs)
{
    temp.prob <- inv.logit(logistic.model$coefficients[1] +
    mtcars$wt[index] * logistic.model$coefficients[2])
    mtcars$am.guess[index] <- ifelse(temp.prob >= 0.5, 1, 0)
    colour <- ifelse(mtcars$am[index] == mtcars$am.guess[index],
                     'black', 'red')
    if(colour == 'red') text(mtcars$wt[index] + 1.5, temp.prob,
                             rownames(mtcars)[index], col=colour)
    points(mtcars$wt[index], temp.prob, col=colour, lwd=10)
}
```



3. plot quadratic fit alongside linear fit
   i. judging visually, does adding a quadratic term make a difference?
   ii. check the details in the help of the AIC function - which of the models provide the better fit according to the AIC values and the residual deviance respectively?
   iii. in your own words, why might it be good to penalise a model like the quadratic model, we just fitted.

```
quad.logistic.model <- glm(am ~ I(wt^2) + wt + 1, data=mtcars,
                           family='binomial')
par(font.lab=2, font.axis=2, cex=1.2)
```

```r
plot(mtcars$wt, logistic.model$fitted.values,
     xlab='Weight (lb/1000)',
     ylab='Probability of having manual transmission',
     main='Logistic regression, linear vs. quadratic', pch=3)
points(mtcars$wt, quad.logistic.model$fitted.values, col='red',
       pch=3)
legend('topright', c('Linear fit', 'Quadratic fit'), pch=3,
       col=c('black', 'red'))
```




```r
# question i
# no, it doesn't make much of a difference

# question II
print(AIC(logistic.model))
```

```
## [1] 23.17608
```

```r
print(AIC(quad.logistic.model))
```

```
## [1] 25.11779
```

```r
# the linear model with only the linear term has the lower AIC, and is thus the better fit according to

# question iii
# adding more parameters always results in a better fit when looking at the residual deviance - we thus

# For example, fitting the perfect model
```

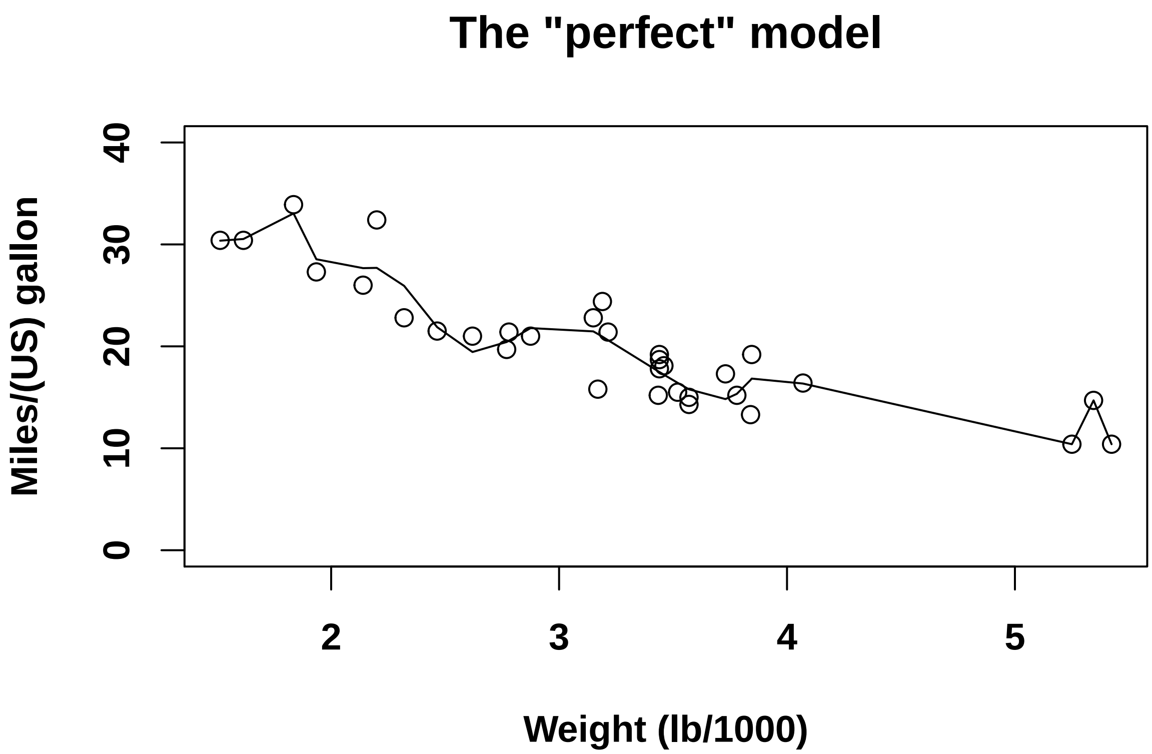
```r
perfect.formula <- 'mpg ~ 1'
dim(mtcars)[1]
```

```
## [1] 32
```

```r
for(index in 2:n.obs)
{
    perfect.formula <- paste(perfect.formula, ' + I(wt^',
                             index-1, ')', sep='')
}
perfect.formula <- as.formula(perfect.formula)
perfect.model <- lm(perfect.formula, data=mtcars)
sort.list. <- sort(mtcars$wt, index.return=TRUE)
plot(sort.list$x,
     perfect.model$fitted.values[sort.list$ix], type='l',
     ylim=c(0, 40), xlab='Weight (lb/1000)',
     ylab='Miles/(US) gallon', main='The "perfect" model')
points(mpg ~ wt, data=mtcars)
## we are not actually getting a perfect fit however...
print(SS.perfect <- sum(perfect.model$residuals^2))
```

```
## [1] 122.8369
```

```r
# This is likely due to lack of numeric precision
library(Matrix)
```




```r
## rank of matrix
print(rankMatrix(model.matrix(perfect.model))[1])
```

```
## [1] 7
```

```
## full rank would be 32
```