Optimization Techniques
Linear and Logistic Regression

James Ponce

March 5, 2019

Introduction

In this study we are going to explain the basis of optimization techniques for two models: the linear regression model and the logistic regression model. For both of these we will cover the implementation of the models in R, after that we will compute the models to show the numeric results. Finally, we will make a result comparison between the model made from scratch in contrast with the results obtained using the R build-in funcions, such as lm (for linear regression) and glm(for logistic regression).

Linear Regression Model

Model

The following equation represents the linear model in matrix notation:

$$y = \beta X + \epsilon$$

Where:

X: is the matrix that represents the independent variables.
y: is the vector that represents the dependent or response variable.
β: is the vector of coefficients.
ϵ: is the vector of errors.

The aim of this model is minimize the square sum of errors, represented by:

$$\sum_{i=1}^{n} \epsilon^2 = \begin{bmatrix} \epsilon_1\epsilon_2\epsilon_3, ...\epsilon_n \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \end{bmatrix} = \epsilon\epsilon^t$$

Replacing $\epsilon = y - x\beta$ in the previous equation, we have:

$$\epsilon\epsilon^t = (y - X\beta)^t(y - X\beta)$$

We calculate the first derivative

$$\frac{\partial(\epsilon\epsilon^t)}{\partial\beta} = \frac{\partial((y - X\beta)^t(y - X\beta))}{\partial\beta}$$

$$\frac{\partial(\epsilon\epsilon^t)}{\partial\beta} = -2X^t(y - X\beta)$$

Given the theory, we make the previous function equal to zero, then solve for β.

$$-2X^t(y - X\beta) = 0$$

Or assuming the normal equation:

$$X^t y = X^t X\beta$$

The solution for obtaining the coefficients, vector β is:

$$\beta = (X^t X)^{-1} X^t y$$

Where:

X^t: is the transpose matrix of X.
X^{-1}: is the inverse matrix of X.

Data

In order to test this model we use the dataset Boston which is included in the MASS R package. The Boston data frame includes 14 variables 506 observations. Among those variables we consider one dependent variable which is the `median value of owner-occupied homes` and 13 independent variables such as `crime rate, tax rate, number of rooms, etc`.

[Boston Dataset]

```
library(MASS)
str(Boston)
```

```
## 'data.frame':    506 obs. of  14 variables:
##  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
##  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
##  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ black  : num  397 397 393 395 397 ...
##  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
##  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

Implementation in R

According to the model in matrix notation, we implement the calculation of the coefficients vector β as follows using matrix operations.

```r
# library(MASS)

# Dependent variable median value of owner-occupied homes
y <- Boston$medv

# Matrix of feature variables from Boston
# Select all variables but the last "medv"
X <- as.matrix(Boston[-ncol(Boston)])

# Vector of ones with same length as rows in Boston
int <- rep(1, length(y))

# Add intercept column to X
X <- cbind(int, X)

# Implement closed-form solution
# Solve function compute the inverse of matrix [t(X)%*%X]
betas <- solve(t(X) %*% X) %*% t(X) %*% y

# Round with 2 decimals
betas <- round(betas, 2)
```

Numeric Results

We compare the results that we have obtained using our function in contrast with the results computed with `lm()` function. The results are similar, its mean that we have the same β coefficientes for the optimized objective function and, of course, the same prediction and error.

Also, we can consider the variable `int` which means `proportion of non-retail business acres per town` has the most positive impact in the `median value of the house`. In the other hand, we realized that the variable `nox` which means `nitrogen oxides concentration` represents the most negative effect to our response variable.

```r
# Comparison face to lm() function
# lm for linear regression model
lm.mod <- lm(medv ~ ., data=Boston)

# Round with 2 decimals
lm.betas <- round(lm.mod$coefficients, 2)

# Create data.frame of results
results <- data.frame(own_function=betas, lm_results=lm.betas)
print(results)
```

```
##         own_function lm_results
## int            36.46      36.46
## crim           -0.11      -0.11
## zn              0.05       0.05
## indus           0.02       0.02
## chas            2.69       2.69
## nox           -17.77     -17.77
## rm              3.81       3.81
## age             0.00       0.00
```

```
## dis           -1.48      -1.48
## rad            0.31       0.31
## tax           -0.01      -0.01
## ptratio       -0.95      -0.95
## black          0.01       0.01
## lstat         -0.52      -0.52
```

Logistic Regression Model

Model

The following equation represents the logistic regression model in matrix notation:

$$\log(\frac{p}{(1-p)}) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \theta_3 X_3 ... + \theta_n X_n$$

The cost function is represented by the following equation:

$$J = \frac{-1}{m} \sum_{i=1}^{m} (y^{(i)} log(h(x^{(i)})) + (1 - y^{(i)}) log(1 - h(x^{(i)}))$$

Where $h(x)$ is the sigmoid function, defined by:

$$h(x) = \frac{1}{1 + e^{-x.\theta}}$$

Data

For testing this logistic regression model, we use the dataset `NBA stats` to predict an outcome (made or not) of a shot based on the shot distance, shot clock remaining and distance to closest defender.

The `NBA stats` data frame used for this study includes 4 variables 203590 observations. Among those variables we consider one dependent variable `FGM` which is the `shot made` and 3 independent variables such as `shot clock remaining, shot distance and distance to closest defender`.

[NBA stats Dataset]

```
# Load the dataset
# library(dplyr)
shot <- read.csv('shot.csv', header = T, stringsAsFactors = F)
shot.df <- shot[,c("FGM", "SHOT_CLOCK", "SHOT_DIST", "CLOSE_DEF_DIST")]
str(shot.df)
```

```
## 'data.frame':    203590 obs. of  4 variables:
##  $ FGM           : int  1 0 1 1 1 0 1 0 1 0 ...
##  $ SHOT_CLOCK    : num  10.6 6.2 5.2 16 1.4 14.1 8.3 10.5 14.3 10 ...
##  $ SHOT_DIST     : num  6.5 4 2.8 8.4 4.5 1.3 1.9 6.9 7.6 6.9 ...
##  $ CLOSE_DEF_DIST: num  2.1 1.9 2.2 4.4 4.4 1.9 2.1 1.4 1.9 2.2 ...
```

Implementation in R

According to the model in matrix notation, we implement three functions: the sigmoid, which is the inverse of the logit function, then we use that function to calculate the cost function. After that, we implement the gradient function. Finally, we compute the logistic regression function that uses the cost function and the gradient function into the optim function to optimize the response variable y.

```r
# Loading package
library(dplyr)

# Sigmoid function, inverse of logit
sigmoid <- function(z){1/(1+exp(-z))}

# Cost function
cost <- function(theta, X, y){
  m <- length(y) # number of training examples

  h <- sigmoid(X%*%theta)
  J <- (t(-y)%*%log(h)-t(1-y)%*%log(1-h))/m
  J
}

# Gradient function
grad <- function(theta, X, y){
  m <- length(y)

  h <- sigmoid(X%*%theta)
  grad <- (t(X)%*%(h - y))/m
  grad
}

logisticReg <- function(X, y){
  # Remove NA rows
  temp <- na.omit(cbind(y, X))
  # Add bias term and convert to matrix
  X <- mutate(temp[, -1], bias =1)
  X <- as.matrix(X[,c(ncol(X), 1:(ncol(X)-1))])
  y <- as.matrix(temp[, 1])
  # Initialize theta
  theta <- matrix(rep(0, ncol(X)), nrow = ncol(X))
  # Use the optim function to perform gradient descent
  costOpti <- optim(matrix(rep(0, 4), nrow = 4), cost, grad, X=X, y=y)
  # Return coefficients
  return(costOpti$par)
}
```

Numeric Results

We compare the results that we have obtained using our function in contrast with the results computed with `glm()` function. The results are similar, its mean that we have the same β coefficientes for the optimized objective function.

```r
# Computing own logisticReg function to calculate coeficients
# Dividing dataset in variables X, y
shot.X <- shot.df[, -1]
shot.y <- shot.df[, 1]

mod <- logisticReg(shot.X, shot.y)
own_function<-as.vector(mod)

# Using glm
mod1 <- glm(as.factor(FGM) ~ SHOT_CLOCK + SHOT_DIST + CLOSE_DEF_DIST,
            family=binomial(link = "logit"), data=shot.df)
lm_results<-as.vector(mod1$coefficients)

# Comparing with lm() function
comparison<-cbind(own_function,lm_results)
rownames(comparison)<-c("(intercept)","SHOT_CLOCK","SHOT_DIST","CLOSE_DEF_DIST")
comparison
```

```
##                own_function  lm_results
## (intercept)     -0.06444830 -0.06663629
## SHOT_CLOCK       0.01886594  0.01889689
## SHOT_DIST       -0.05952669 -0.05942813
## CLOSE_DEF_DIST   0.10610004  0.10605534
```

References

(Pardoe 2018) (Ripley 2018) (Barnes 2006) (Ma. 2016) (Ma 2015)

Barnes, Randal J. 2006. "Matrix Differentiation." August. https://atmos.washington.edu/~dennis/MatrixCalculus.pdf.

Ma, Jun. 2015. "NBA Statistics." December. https://github.com/JunWorks/NBAstat.

Ma., Jun. 2016. "Logistic Regression from Scratch." February. https://rpubs.com/junworks/Understanding-Logistic-Regression-from-Scratch.

Pardoe, Iain. 2018. "A Matrix Formulation of the Multiple Regression Model." https://newonlinecourses.science.psu.edu/stat501/node/382/.

Ripley, Brian. 2018. "Housing Values in Suburbs of Boston." https://stat.ethz.ch/R-manual/R-devel/library/MASS/html/Boston.html.