Portfolio#2 - function

Ruinan(Victor) Zhang

22 October 2016

# Background

Given the past launching data, the task requires to implement logistic regression to predict the possible failure of O-rings based on Launch Temeprature, and Leak-Check Pressure.

# Approach

To implement the logistic regression, an update\_beta() function was first implemented to update the values of beta in every iteration. The beta at each iteration is calculated as: . Once the update size gets smaller than 1e-07, the algorithm is stopped and we have a converged output logistic regression model. If the algorithm does not stop after 250 iterations, an error message will automatically be raised and stopped the algorithm.

### result

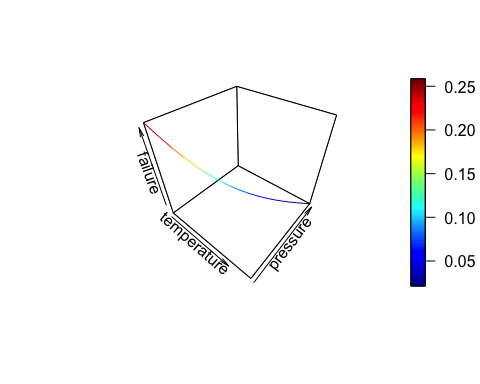
After importing the data and running the logistic regression on the dataset, the algorithm converged, and output the result. With the given situation of temperature being 31 degrees F and leak-check pressure being 200 psi, the output indicates there is a strong high probability that the mission is going to fail.

## Loading required package: MASS

number\_failure <- c(0,1,rep(0,6),1,1,1,0,0,1,rep(0,6),1,0,1)  
launch\_temp <- c(66,70,69,68,67,72,73,70,57,63,70,78,67,53,67,75,70,81,76,79,75,76,58)  
leak\_check\_pressure <- c(rep(50,6),rep(100,2),rep(200,15))  
X\_train <- cbind(launch\_temp, leak\_check\_pressure) # set training design matrix  
  
beta <- lrm(X\_train,number\_failure) # calculate beta  
x<-c(1,31,200)  
exp(sum(beta \* x))/(1+exp(sum(beta\*x))) # calcualte output response

## [1] 0.9997421

As it is shown in the plot below, the launch is mission is extremely likely to fail when the temperature is low and check-leak pressure is high.



# Reflection

However, the output of the algorithm in this particular case is extremely close to 1 which looks like it is highly possible to be a failture under given situation. A confidence level should be checked for future development to make the output more convincing. One should pay particular attention to index during programming this algorithm since R's default starts from 1, and the logistic regression makes more sense to have a initial response with index 0. Therefore, the iteration with parameter k is actually the (k-1)th iteration.

# Full Code

################################################################################  
#Program: PortfolioFunctions Class: MA386 - Statistical Programming   
#Description: a logistic regression was implemented to predict the possible failure   
# of an upcoming space shuttle luanch  
#Author: Ruinan(Victor) Zhang Date: 09/19/2016 Modified: 09/19/2016  
#  
#Notes:   
  
################################################################################  
  
# load libraries and packages  
library(Matrix)  
require(MASS)  
library(plot3D)  
  
# function lrm takes a design matrix and response vector  
lrm <- function(X\_origin,y){  
 X <- cbind(rep(1,nrow(X\_origin)),X\_origin)  
 n = ncol(X)  
 k = 1 # count for number of iterations  
   
 p <- list() # all intermediate result was stored in a list  
 beta = list()  
 beta[[k]] <- rep(0,n) # betas are started with zeros  
 p[[k]] <- exp(X %\*% beta[[k]])/(1+exp(X %\*% beta[[k]])) # initialize  
   
 update\_beta <- function(k,beta,p){  
 #browser()  
 if (k >250){ # if the iteration is beyond 250, it's probablity not going to converge  
 # raise error if not converge  
 stop("algorithm does not converge after 250 iteration")  
 }  
 D <- Diagonal(length(as.vector(p[[k]])),as.vector(p[[k]]))  
 tem <- t(X) %\*% D %\*% X # this is a temperary vector variable  
 tem <- matrix(tem,n,n)  
 delta <- ginv(tem) %\*% t(X) %\*% (y-p[[k]]) # update step  
   
 criteria<- sum(delta^2)/sum(beta[[k]]^2) # converging criteria  
 if (criteria < 10^(-7)){  
 return(beta[[k]])  
 }  
 beta[[k+1]] <- beta[[k]]+delta  
 p[[k+1]] <- exp(X %\*% beta[[k+1]])/(1+exp(X %\*% beta[[k+1]]))  
 k=k+1  
 update\_beta(k,beta,p)  
 }  
 beta\_final <- update\_beta(k,beta,p)  
   
 return(beta\_final)  
}  
  
number\_failure <- c(0,1,rep(0,6),1,1,1,0,0,1,rep(0,6),1,0,1)  
launch\_temp <- c(66,70,69,68,67,72,73,70,57,63,70,78,67,53,67,75,70,81,76,79,75,76,58)  
leak\_check\_pressure <- c(rep(50,6),rep(100,2),rep(200,15))  
X\_train <- cbind(launch\_temp, leak\_check\_pressure) # set training design matrix  
  
beta <- lrm(X\_train,number\_failure) # calculate beta  
x<-c(1,31,200)  
exp(sum(beta \* x))/(1+exp(sum(beta\*x))) # calcualte output response  
  
temp = seq(65,84)  
pres = seq(50,200,length.out = 20)  
intercept <- rep(1,20)  
x<- cbind(intercept,temp,pres)  
z <- exp((x %\*% beta))/(1+exp((x %\*% beta)))  
  
lines3D(temp,pres,z,xlab="temperature",ylab="pressure", zlab="failure")