Assignment 2

Solichatus Zahroh

2022-10-22

## PART I: Programming your own t-test function

1. Create the t-test function

MyTtest <- function(x1,x2){  
 diff.mean <- mean(x1)-mean(x2) #calculate the mean difference  
 n1 <- length(x1) #number of dataset of the 1st group  
 n2 <- length(x2) #number of dataset of the 2nd group  
 s1 <- (n1-1)\*(sd(x1))^2 #calculate std deviation of the 1st group  
 s2 <- (n2-1)\*(sd(x2))^2 #calculate std deviation of the 2nd group  
 s <- sqrt((s1+s2)/(n1+n2-2)) #calculate estimate std deviation  
 t <-diff.mean/(s\*sqrt(1/n1+1/n2)) #produce statistic value (t)  
 return(t)  
}

Here, the function that has output of statistic value of T-student is defined and can be used to show the differences between group means.

1. Test your function with data and compare with built-in t-test function

CSFI <- c(2,5,5,6,6,7,8,9) #the first group  
TFI <- c(1,1,2,3,3,4,5,7,7,8) #the second group  
t <- t.test(CSFI,TFI,var.equal = T)$statistic #t value output from the built-in function  
t.func <- MyTtest(CSFI,TFI) #t-value output from my function  
a <- list(t.built=t,t.func=t.func)  
names(a) <- c("t.built-in","t.function")  
a

## $`t.built-in`  
## t   
## 1.680507   
##   
## $t.function  
## [1] 1.680507

#the function produce the same value with the built-in function

The result of the built-in function is 1.680507 which has the same result with my function.

1. Calculate p-value (two sided test) of the t-test and compare with built-in function

MyTtest <- function(x1,x2){  
 diff.mean <- mean(x1)-mean(x2) #calculate the mean difference  
 n1 <- length(x1) #number of dataset of the 1st group  
 n2 <- length(x2) #number of dataset of the 2nd group  
 s1 <- (n1-1)\*(sd(x1))^2 #calculate std deviation of the 1st group  
 s2 <- (n2-1)\*(sd(x2))^2 #calculate std deviation of the 2nd group  
 s <- sqrt((s1+s2)/(n1+n2-2)) #calculate estimate std deviation  
 t <- diff.mean/(s\*sqrt(1/n1+1/n2)) #produce statistic value (t)  
 p.value <- 2\*pt(t,n1+n2-2,lower.tail = F)   
 result <- list(t,p.value)  
 names(result) <- c("t-statistic","two-sided p-value")  
 return(result)  
}  
MyTtest(CSFI,TFI)

## $`t-statistic`  
## [1] 1.680507  
##   
## $`two-sided p-value`  
## [1] 0.1122736

built.func <- t.test(CSFI,TFI,var.equal = T)  
func <- list(built.func$statistic,built.func$p.value)  
names(func) <- c("t-statistic","two-sided p-value")  
func

## $`t-statistic`  
## t   
## 1.680507   
##   
## $`two-sided p-value`  
## [1] 0.1122736

The two-sided p-value of the built-in function is 0.1122736 and the result of my function is also 0.1122736. Both results are the same for the t-statistic and two-sided p-value.

1. Organize the output in the form of a list

MyTtest <- function(x1,x2){  
 diff.mean <- mean(x1)-mean(x2) #calculate the mean difference  
 n1 <- length(x1) #number of dataset of the 1st group  
 n2 <- length(x2) #number of dataset of the 2nd group  
 s1 <- (n1-1)\*(sd(x1))^2 #calculate std deviation of the 1st group  
 s2 <- (n2-1)\*(sd(x2))^2 #calculate std deviation of the 2nd group  
 s <- sqrt((s1+s2)/(n1+n2-2)) #calculate estimate std deviation  
 t <-diff.mean/(s\*sqrt(1/n1+1/n2)) #produce statistic value (t)  
 p.value <- 2\*pt(abs(t),n1+n2-2,lower.tail = F)   
 result <- list(t,p.value)  
 names(result) <- c("t-statistic","two-sided p-value")  
 return(result)  
}  
MyTtest(CSFI,TFI)

## $`t-statistic`  
## [1] 1.680507  
##   
## $`two-sided p-value`  
## [1] 0.1122736

MyTtest(TFI,CSFI)

## $`t-statistic`  
## [1] -1.680507  
##   
## $`two-sided p-value`  
## [1] 0.1122736

Here is the list of the function, consists of statistics of T-student and the two-sided p-value.

## PART 2: Multiple regression analysis and matrix algebra

The data for this exercise are about the number of species of tortoise on the various Galapagos Islands. There are 30 cases (islands) and 7 variables in the dataset. The variables are: • Species: the number of species of tortoise found on the island • Endemics: the number of endemic species on the island • Area: the area of the island (km2) • Elevation: the highest elevation of the island (m) • Nearest: the distance from the nearest island (km) • Scruz: the distance from Santa Cruz island (km) • Adjacent: the area of the adjacent island (km2)

Species is the dependent variable and only the variables Area,Elevation,Endemics are used as predictors.

1. Use the in-built R-function for linear regression and the summary() function to obtain the output for a multiple linear regression analysis on the variables described. Obtain predicted values and residuals using the extractor functions. Make a plot of the predicted values against the residuals. Give an interpretation of the results of the regression model and the plot.

gala<- read.csv("~/Files/Utrecht!/Course First Semester/Computational inference with R/gala(2).txt", sep="")  
str(gala)

## 'data.frame': 30 obs. of 7 variables:  
## $ Species : int 58 31 3 25 2 18 24 10 8 2 ...  
## $ Endemics : int 23 21 3 9 1 11 0 7 4 2 ...  
## $ Area : num 25.09 1.24 0.21 0.1 0.05 ...  
## $ Elevation: int 346 109 114 46 77 119 93 168 71 112 ...  
## $ Nearest : num 0.6 0.6 2.8 1.9 1.9 8 6 34.1 0.4 2.6 ...  
## $ Scruz : num 0.6 26.3 58.7 47.4 1.9 ...  
## $ Adjacent : num 1.84 572.33 0.78 0.18 903.82 ...

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.3.6 ✔ purrr 0.3.4   
## ✔ tibble 3.1.8 ✔ dplyr 1.0.10  
## ✔ tidyr 1.2.1 ✔ stringr 1.4.1   
## ✔ readr 2.1.2 ✔ forcats 0.5.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

gala <- gala %>%  
 select(Species,Area,Elevation,Endemics)  
summary(gala) #show descriptive statistics of the dataset

## Species Area Elevation Endemics   
## Min. : 2.00 Min. : 0.010 Min. : 25.00 Min. : 0.00   
## 1st Qu.: 13.00 1st Qu.: 0.258 1st Qu.: 97.75 1st Qu.: 7.25   
## Median : 42.00 Median : 2.590 Median : 192.00 Median :18.00   
## Mean : 85.23 Mean : 261.709 Mean : 368.03 Mean :26.10   
## 3rd Qu.: 96.00 3rd Qu.: 59.237 3rd Qu.: 435.25 3rd Qu.:32.25   
## Max. :444.00 Max. :4669.320 Max. :1707.00 Max. :95.00

model.gala <- lm(Species~Area+Elevation+Endemics,data=gala) #define model for the multiple regression analysis  
summary(model.gala) #show the coefficient, std error, t value and p-value for each variable. R-square and adj R-square can also be seen here

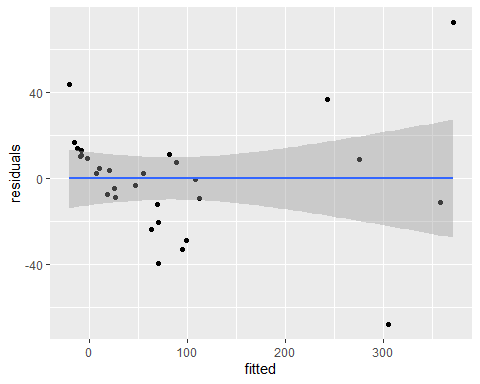
##   
## Call:  
## lm(formula = Species ~ Area + Elevation + Endemics, data = gala)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -67.691 -10.530 2.387 10.529 72.723   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -15.891237 7.569210 -2.099 0.0456 \*   
## Area 0.012669 0.008936 1.418 0.1681   
## Elevation -0.041439 0.023653 -1.752 0.0916 .   
## Endemics 4.331791 0.304686 14.217 8.97e-14 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 27.29 on 26 degrees of freedom  
## Multiple R-squared: 0.9492, Adjusted R-squared: 0.9433   
## F-statistic: 161.8 on 3 and 26 DF, p-value: < 2.2e-16

list\_model <- list(model.gala$coefficients,model.gala$fitted.values,model.gala$residuals) #store the coefficients, fitted values, and residuals as a list  
names(list\_model) <- c("coefficients", "fitted values", "residuals")  
list\_model ##show coefficients, fitted values, and residuals using built-in function

## $coefficients  
## (Intercept) Area Elevation Endemics   
## -15.89123674 0.01266946 -0.04143887 4.33179106   
##   
## $`fitted values`  
## Baltra Bartolome Caldwell Champion Coamano Daphne.Major   
## 69.719986 70.575249 -7.617234 21.189962 -14.749605 26.831547   
## Daphne.Minor Darwin Eden Enderby Espanola Fernandina   
## -19.744038 7.499091 -1.505852 -11.866527 89.268685 81.850429   
## Gardner1 Gardner2 Genovesa Isabela Marchena Onslow   
## 55.725928 -7.960813 63.483255 358.059788 71.166995 -8.263500   
## Pinta Pinzon Las.Plazas Rabida SanCristobal SanSalvador   
## 112.941626 108.306284 19.202543 98.916385 242.993682 304.691338   
## SantaCruz SantaFe SantaMaria Seymour Tortuga Wolf   
## 371.276645 94.971327 275.974100 47.349219 11.071173 25.642331   
##   
## $residuals  
## Baltra Bartolome Caldwell Champion Coamano Daphne.Major   
## -11.7199865 -39.5752492 10.6172338 3.8100381 16.7496050 -8.8315474   
## Daphne.Minor Darwin Eden Enderby Espanola Fernandina   
## 43.7440378 2.5009091 9.5058520 13.8665272 7.7313153 11.1495707   
## Gardner1 Gardner2 Genovesa Isabela Marchena Onslow   
## 2.2740715 12.9608131 -23.4832547 -11.0597884 -20.1669948 10.2634996   
## Pinta Pinzon Las.Plazas Rabida SanCristobal SanSalvador   
## -8.9416260 -0.3062841 -7.2025433 -28.9163846 37.0063185 -67.6913383   
## SantaCruz SantaFe SantaMaria Seymour Tortuga Wolf   
## 72.7233548 -32.9713271 9.0258998 -3.3492186 4.9288274 -4.6423306

var.y <- as.data.frame(model.gala$fitted.values) #change format of fitted values to dataframe  
var.x <- as.data.frame(model.gala$residuals) #change format of residuals to dataframe  
comb <- as.data.frame(cbind(var.x,var.y)) #combine both values  
colnames(comb) <- c("residuals", "fitted") #rename the columns  
ggplot(comb, aes(x=fitted,y=residuals)) + geom\_point() + geom\_smooth(method = "lm") #generate plot of fitted values against residuals

## `geom\_smooth()` using formula 'y ~ x'

 Multiple regression can be done using lm function. The result consists of the coefficients, standard error, t value, and the p-value. The fitted values or predicted values and residuals of the model also produced. The regression equation of this model is yhat = -15.89 + 0.01Area - 0.04Elevation + 4.33Endemics. However, based on alpha=5%, only endemics variable is significant then the model is yhat = -15.89 + 4.33Endemics because if the area and elevation variaables are included, the variance explained by the model is not increase at all. The plot of fitted values against residuals can be useful to determine whether some assumptions are met or not. 1. The linearity assumption is met because the blue line is zero (indicated by the mean residual value for every fitted value region being close to 0). The model used is lm, the line also lm function. 2. The homoskedasticity assumption holds because the spread of residuals approximately the same across the x-axis. 3. Outliers assumptions holds. Some extreme residuals far from the rest, however it should be inspected whether the outliers are implausible or impossible.

1. Obtain the same estimates as in the previous step, but now with matrix algebra Here is the list with estimates and how to obtain them with matrix algebra: • the regression coefficients βˆ = (X’X)^−1(X’y) • the predicted values yhat = Xβ • the residuals e (easy to obtain once you have the predicted values).

regres <- function(x,y){  
 x <- as.matrix(x) #create matrix of independent variables  
 y <- as.matrix(y) #create matrix of dependent variable  
 mat.a <- solve(t(x)%\*%x) #calculate matrix (X'X)  
 mat.b <- t(x)%\*%y #calculate matrix (X'Y)  
 beta <- mat.a%\*%mat.b #calculate coefficients  
 yhat <- x%\*%beta #calculate fitted values  
 resid <- y-x%\*%beta #calculate residuals  
 my\_list <- list(beta,yhat,resid)  
 names(my\_list) <- c("coefficients", "fitted values", "residuals")  
 return(my\_list)  
}  
x <- gala %>%   
 mutate(Intercept=1) %>% #adding intercept  
 dplyr::select(Intercept,Area,Elevation,Endemics) #independent variables  
y <- gala %>% dplyr::select(Species) #dependent variable  
regres(x,y) #show coefficients, fitted values, and residuals using function

## $coefficients  
## Species  
## Intercept -15.89123674  
## Area 0.01266946  
## Elevation -0.04143887  
## Endemics 4.33179106  
##   
## $`fitted values`  
## Species  
## Baltra 69.719986  
## Bartolome 70.575249  
## Caldwell -7.617234  
## Champion 21.189962  
## Coamano -14.749605  
## Daphne.Major 26.831547  
## Daphne.Minor -19.744038  
## Darwin 7.499091  
## Eden -1.505852  
## Enderby -11.866527  
## Espanola 89.268685  
## Fernandina 81.850429  
## Gardner1 55.725928  
## Gardner2 -7.960813  
## Genovesa 63.483255  
## Isabela 358.059788  
## Marchena 71.166995  
## Onslow -8.263500  
## Pinta 112.941626  
## Pinzon 108.306284  
## Las.Plazas 19.202543  
## Rabida 98.916385  
## SanCristobal 242.993682  
## SanSalvador 304.691338  
## SantaCruz 371.276645  
## SantaFe 94.971327  
## SantaMaria 275.974100  
## Seymour 47.349219  
## Tortuga 11.071173  
## Wolf 25.642331  
##   
## $residuals  
## Species  
## Baltra -11.7199865  
## Bartolome -39.5752492  
## Caldwell 10.6172338  
## Champion 3.8100381  
## Coamano 16.7496050  
## Daphne.Major -8.8315474  
## Daphne.Minor 43.7440378  
## Darwin 2.5009091  
## Eden 9.5058520  
## Enderby 13.8665272  
## Espanola 7.7313153  
## Fernandina 11.1495707  
## Gardner1 2.2740715  
## Gardner2 12.9608131  
## Genovesa -23.4832547  
## Isabela -11.0597884  
## Marchena -20.1669948  
## Onslow 10.2634996  
## Pinta -8.9416260  
## Pinzon -0.3062841  
## Las.Plazas -7.2025433  
## Rabida -28.9163846  
## SanCristobal 37.0063185  
## SanSalvador -67.6913383  
## SantaCruz 72.7233548  
## SantaFe -32.9713271  
## SantaMaria 9.0258998  
## Seymour -3.3492186  
## Tortuga 4.9288274  
## Wolf -4.6423306

The result of the built-in function and my function have the same output. For the independent variables, I firstly add intercept before doing computation because the default of the built-in function has intercept. Another way, if the intercept is not included to the independent variables, the built in function should define without intercept too (lm(y~x-1) or lm(y~0+x)).

1. Write a function that has as input a data set suitable for multiple linear regression. The function should perform the following: • obtain useful descriptive statistics of the data (you are free to choose which statistics as long as it makes sense). You can use matrix algebra here, but it is not necessary. • obtain the regression coefficients, predicted values and residuals using matrix algebra • a simple plot of the predicted values against the residuals. Add comments to your code and write the function as a general function that could be applied to each possible data set with varying numbers of predictors.

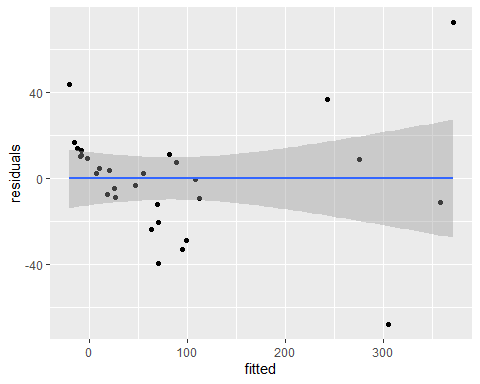
regres.func <- function(x,y){  
 #descriptive statistics  
 all <-as.data.frame(cbind(x,y)) #combine independent and dependent variables  
 all[is.na(all)] = 0  
 mean.v <- sapply(all,mean) #calculate mean of each variable  
 max.v <- sapply(all, max) #calculate maximum value of each variable  
 min.v <- sapply(all, min) #calculate minimum value of each variable  
 desc <- list(mean.v,max.v,min.v) #store those values as a list  
 names(desc) <- c("mean.values", "max.values", "min,values") #rename the list  
   
 #regression   
 x <- as.matrix(x) #create matrix of independent variables  
 y <- as.matrix(y) #create matrix of dependent variable  
 x[is.na(x)] = 0; y[is.na(y)] = 0  
 mat.a <- solve(t(x)%\*%x) #calculate matrix (X'X)  
 mat.b <- t(x)%\*%y #calculate matrix (X'Y)  
 beta <- mat.a%\*%mat.b #calculate coefficients  
 yhat <- x%\*%beta #calculate fitted values  
 resid <- y-x%\*%beta #calculate residuals  
 my\_list <- list(beta,yhat,resid) #store the coefficients, fitted values, and residuals as a list  
 names(my\_list) <- c("coefficients", "fitted.values", "residuals") #rename the list  
   
 result <- list(desc,my\_list) #store the descriptive statistics and the regression outputs as a list  
 names(result) <- c("descriptive statistics","regression") #rename the list  
   
 #plot  
 comb <- as.data.frame(cbind(resid,yhat)) #combine predicted values and residuals  
 colnames(comb) <- c("residuals", "fitted") #rename the column  
 show(ggplot(comb, aes(x=fitted,y=residuals)) + geom\_point() + geom\_smooth(method = "lm")) #generate the plot through ggplot  
   
 return(result)  
}

This function can be applied to all multiple regression models as long as the dataset is a numerical data. If the independent variables are not numerical data, it should be converted to numerical data (dummy variables) because the multiple regression only works for numerical data only. If the dependent variables are not numerical data, another analysis should be used, namely logistic regression. If there are missing values in the dataset, then the missing values need to be removed before the analysis. Otherwise, the NA values converted to 0.

1. Run your function on the tortoise data and show that the results are the same as you obtained in II.1.

x <- gala %>%   
 mutate(Intercept=1) %>% #adding intercept  
 dplyr::select(Intercept,Area,Elevation,Endemics) #independent variables  
y <- gala %>% dplyr::select(Species) #dependent variable  
regres.func(x,y) #show descriptive statistics (mean, max, min values), coefficients, fitted values, and residuals using function, along with the plot.

## `geom\_smooth()` using formula 'y ~ x'



## $`descriptive statistics`  
## $`descriptive statistics`$mean.values  
## Intercept Area Elevation Endemics Species   
## 1.00000 261.70867 368.03333 26.10000 85.23333   
##   
## $`descriptive statistics`$max.values  
## Intercept Area Elevation Endemics Species   
## 1.00 4669.32 1707.00 95.00 444.00   
##   
## $`descriptive statistics`$`min,values`  
## Intercept Area Elevation Endemics Species   
## 1.00 0.01 25.00 0.00 2.00   
##   
##   
## $regression  
## $regression$coefficients  
## Species  
## Intercept -15.89123674  
## Area 0.01266946  
## Elevation -0.04143887  
## Endemics 4.33179106  
##   
## $regression$fitted.values  
## Species  
## Baltra 69.719986  
## Bartolome 70.575249  
## Caldwell -7.617234  
## Champion 21.189962  
## Coamano -14.749605  
## Daphne.Major 26.831547  
## Daphne.Minor -19.744038  
## Darwin 7.499091  
## Eden -1.505852  
## Enderby -11.866527  
## Espanola 89.268685  
## Fernandina 81.850429  
## Gardner1 55.725928  
## Gardner2 -7.960813  
## Genovesa 63.483255  
## Isabela 358.059788  
## Marchena 71.166995  
## Onslow -8.263500  
## Pinta 112.941626  
## Pinzon 108.306284  
## Las.Plazas 19.202543  
## Rabida 98.916385  
## SanCristobal 242.993682  
## SanSalvador 304.691338  
## SantaCruz 371.276645  
## SantaFe 94.971327  
## SantaMaria 275.974100  
## Seymour 47.349219  
## Tortuga 11.071173  
## Wolf 25.642331  
##   
## $regression$residuals  
## Species  
## Baltra -11.7199865  
## Bartolome -39.5752492  
## Caldwell 10.6172338  
## Champion 3.8100381  
## Coamano 16.7496050  
## Daphne.Major -8.8315474  
## Daphne.Minor 43.7440378  
## Darwin 2.5009091  
## Eden 9.5058520  
## Enderby 13.8665272  
## Espanola 7.7313153  
## Fernandina 11.1495707  
## Gardner1 2.2740715  
## Gardner2 12.9608131  
## Genovesa -23.4832547  
## Isabela -11.0597884  
## Marchena -20.1669948  
## Onslow 10.2634996  
## Pinta -8.9416260  
## Pinzon -0.3062841  
## Las.Plazas -7.2025433  
## Rabida -28.9163846  
## SanCristobal 37.0063185  
## SanSalvador -67.6913383  
## SantaCruz 72.7233548  
## SantaFe -32.9713271  
## SantaMaria 9.0258998  
## Seymour -3.3492186  
## Tortuga 4.9288274  
## Wolf -4.6423306

The result of this function consists of 3 descriptive statistics (mean, max, and min values), the coefficient regression, the fitted values or predicted value, and the residuals. Also, the plot from ggplot with regression line with lm function is generated. This function has the same output as the built-in function in II.1.