Simple Regression Linear

Agnes Septilia

5/11/2021

library(tidyverse)

## 

## INTRODUCTION

In this program we will learn about how to make Simple Linear Regression, by breaking down to each formula.

First, we will start from making the dataset dummy

# define x value as predictor  
predictor <- c(15, 20, 25, 37, 40, 45, 48, 50, 55, 61, 64, 67, 70)  
  
# define y value as target  
target <- c(100, 135, 135, 150, 250, 270, 290, 360, 375, 400, 500, 600, 700)  
  
# set as dataframe  
df <- data.frame(x = predictor, y = target)  
df

## x y  
## 1 15 100  
## 2 20 135  
## 3 25 135  
## 4 37 150  
## 5 40 250  
## 6 45 270  
## 7 48 290  
## 8 50 360  
## 9 55 375  
## 10 61 400  
## 11 64 500  
## 12 67 600  
## 13 70 700

**PART 1 : FIND THE REGRESSION FORMULA**

In Linear Regression, the first information we have to check is the prediction formula : y = a + bx

# Start with supporting variable  
df <- df %>%   
 mutate(xy = x \* y,  
 x\_sq = x \*\* 2,  
 y\_sq = y \*\* 2)  
n <- nrow(df) # amount of predictor   
  
# assign `a` value  
a <- (sum(df$y) \* sum(df$x\_sq) - sum(df$x) \* sum(df$xy)) /   
 (n \* sum(df$x\_sq) - (sum(df$x))\*\*2)  
  
  
# assign `b` value  
b <- (n \* sum(df$xy) - sum(df$x) \* sum(df$y)) /  
 (n \* sum(df$x\_sq) - (sum(df$x))\*\*2)  
  
paste(sprintf("The formula is y = %.3f + %.3fx", a, b))

## [1] "The formula is y = -118.420 + 9.723x"

We have got the formula: y = -118.420 + 9.723x. Now we calculate the predicted y using this formula

df$y\_pred <- a + (b \* df$x)  
df

## x y xy x\_sq y\_sq y\_pred  
## 1 15 100 1500 225 10000 27.42085  
## 2 20 135 2700 400 18225 76.03439  
## 3 25 135 3375 625 18225 124.64794  
## 4 37 150 5550 1369 22500 241.32044  
## 5 40 250 10000 1600 62500 270.48857  
## 6 45 270 12150 2025 72900 319.10211  
## 7 48 290 13920 2304 84100 348.27024  
## 8 50 360 18000 2500 129600 367.71566  
## 9 55 375 20625 3025 140625 416.32920  
## 10 61 400 24400 3721 160000 474.66546  
## 11 64 500 32000 4096 250000 503.83359  
## 12 67 600 40200 4489 360000 533.00171  
## 13 70 700 49000 4900 490000 562.16984

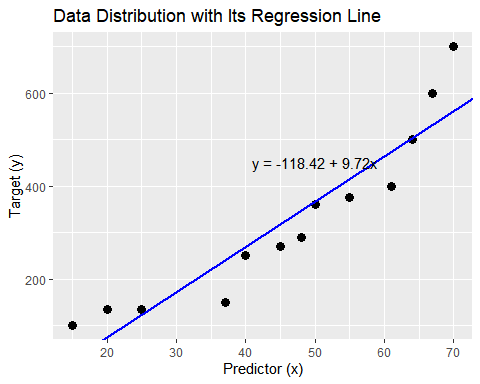
We will check the R-squared value. This is to check whether the linear regression model will be the good fit for the data

r <- (n \* sum(df$xy) - sum(df$x) \* sum(df$y)) /   
 sqrt ((n \* sum(df$x\_sq) - (sum(df$x))\*\*2) \* (n \* sum(df$y\_sq) - (sum(df$y))\*\*2))  
  
r\_sq <- r \*\* 2 # this value is called Multiple R Squared  
  
# Meanwhile, adjusted R-squared will be as follow  
k <- 1 # we only have one independent variable  
adjusted\_r\_sq <- 1 - (((1 - r\_sq) \* (n - 1)) / (n-k-1))  
  
paste(sprintf("The model fits the data with percentage %.2f%%", adjusted\_r\_sq\*100))

## [1] "The model fits the data with percentage 85.89%"

As the end of PART 1, let’s see how the distribution of data, include with the regression line (predicted value line)

df %>%   
 ggplot(aes(x = x, y = y)) +  
 geom\_point(size = 3) +  
 geom\_abline(intercept = a, slope = b, size = 1, color = "blue") +   
 labs(title = "Data Distribution with Its Regression Line",   
 x = "Predictor (x)",  
 y = "Target (y)") +  
 annotate(geom = "text", x = 50, y = 450, label = "y = -118.42 + 9.72x")



**PART 2 : FIND THE RESIDUAL**

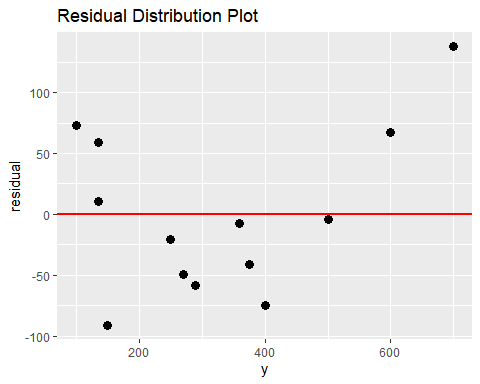
Residual is the discrepancy between actual y value with predicted y value

df$residual <- df$y - df$y\_pred  
df

## x y xy x\_sq y\_sq y\_pred residual  
## 1 15 100 1500 225 10000 27.42085 72.579155  
## 2 20 135 2700 400 18225 76.03439 58.965610  
## 3 25 135 3375 625 18225 124.64794 10.352065  
## 4 37 150 5550 1369 22500 241.32044 -91.320443  
## 5 40 250 10000 1600 62500 270.48857 -20.488570  
## 6 45 270 12150 2025 72900 319.10211 -49.102115  
## 7 48 290 13920 2304 84100 348.27024 -58.270242  
## 8 50 360 18000 2500 129600 367.71566 -7.715660  
## 9 55 375 20625 3025 140625 416.32920 -41.329205  
## 10 61 400 24400 3721 160000 474.66546 -74.665459  
## 11 64 500 32000 4096 250000 503.83359 -3.833586  
## 12 67 600 40200 4489 360000 533.00171 66.998287  
## 13 70 700 49000 4900 490000 562.16984 137.830161

In Linear Regression analysis, we will check the distribution of Residual itself

df %>%   
 ggplot(aes(x = y, y = residual)) +  
 geom\_point(size = 3) +   
 geom\_hline(yintercept = 0, size = 1, color = "red") +   
 labs(title = "Residual Distribution Plot")



**PART 3 : FIND THE STANDARDIZED RESIDUAL**

Before we check the standardized residual, there are several supporting variable we need to make

df$residual\_sq <- df$residual \*\* 2  
  
predictor\_mean <- mean(df$x)  
df$predictor\_sd <- (df$x - predictor\_mean) \*\* 2 # deviation of predictor data  
predictor\_ssdev <- sum(df$predictor\_sd) # sum of square of predictor standard deviation   
  
RSE <- sqrt(sum(df$residual\_sq) / (n-k-1))

We start by counting the leverage; which by definition, is how far an observation value, from those of the other observations.

df$leverage <- (1/n) + (((df$x - predictor\_mean) \*\* 2) / predictor\_ssdev)  
df

## x y xy x\_sq y\_sq y\_pred residual residual\_sq predictor\_sd  
## 1 15 100 1500 225 10000 27.42085 72.579155 5267.73372 956.236686  
## 2 20 135 2700 400 18225 76.03439 58.965610 3476.94316 672.005917  
## 3 25 135 3375 625 18225 124.64794 10.352065 107.16525 437.775148  
## 4 37 150 5550 1369 22500 241.32044 -91.320443 8339.42329 79.621302  
## 5 40 250 10000 1600 62500 270.48857 -20.488570 419.78149 35.082840  
## 6 45 270 12150 2025 72900 319.10211 -49.102115 2411.01768 0.852071  
## 7 48 290 13920 2304 84100 348.27024 -58.270242 3395.42107 4.313609  
## 8 50 360 18000 2500 129600 367.71566 -7.715660 59.53140 16.621302  
## 9 55 375 20625 3025 140625 416.32920 -41.329205 1708.10316 82.390533  
## 10 61 400 24400 3721 160000 474.66546 -74.665459 5574.93071 227.313609  
## 11 64 500 32000 4096 250000 503.83359 -3.833586 14.69638 326.775148  
## 12 67 600 40200 4489 360000 533.00171 66.998287 4488.77052 444.236686  
## 13 70 700 49000 4900 490000 562.16984 137.830161 18997.15314 579.698225  
## leverage  
## 1 0.32446533  
## 2 0.25088614  
## 3 0.19025051  
## 4 0.09753475  
## 5 0.08600502  
## 6 0.07714365  
## 7 0.07803975  
## 8 0.08122586  
## 9 0.09825162  
## 10 0.13576805  
## 11 0.16151579  
## 12 0.19192321  
## 13 0.22699032

Then, we calculate the Standardized Residuals

df$residual\_std <- df$residual / (RSE \* sqrt(1 - df$leverage))  
df

## x y xy x\_sq y\_sq y\_pred residual residual\_sq predictor\_sd  
## 1 15 100 1500 225 10000 27.42085 72.579155 5267.73372 956.236686  
## 2 20 135 2700 400 18225 76.03439 58.965610 3476.94316 672.005917  
## 3 25 135 3375 625 18225 124.64794 10.352065 107.16525 437.775148  
## 4 37 150 5550 1369 22500 241.32044 -91.320443 8339.42329 79.621302  
## 5 40 250 10000 1600 62500 270.48857 -20.488570 419.78149 35.082840  
## 6 45 270 12150 2025 72900 319.10211 -49.102115 2411.01768 0.852071  
## 7 48 290 13920 2304 84100 348.27024 -58.270242 3395.42107 4.313609  
## 8 50 360 18000 2500 129600 367.71566 -7.715660 59.53140 16.621302  
## 9 55 375 20625 3025 140625 416.32920 -41.329205 1708.10316 82.390533  
## 10 61 400 24400 3721 160000 474.66546 -74.665459 5574.93071 227.313609  
## 11 64 500 32000 4096 250000 503.83359 -3.833586 14.69638 326.775148  
## 12 67 600 40200 4489 360000 533.00171 66.998287 4488.77052 444.236686  
## 13 70 700 49000 4900 490000 562.16984 137.830161 18997.15314 579.698225  
## leverage residual\_std  
## 1 0.32446533 1.25730849  
## 2 0.25088614 0.97001543  
## 3 0.19025051 0.16379680  
## 4 0.09753475 -1.36869452  
## 5 0.08600502 -0.30513604  
## 6 0.07714365 -0.72775787  
## 7 0.07803975 -0.86406116  
## 8 0.08122586 -0.11460998  
## 9 0.09825162 -0.61968092  
## 10 0.13576805 -1.14355838  
## 11 0.16151579 -0.05960895  
## 12 0.19192321 1.06118511  
## 13 0.22699032 2.23205843

In R, we can plot standardized residual with qqplot directly. However, here we want to know where the calculation is from. Let’s start by making another dataset.

# sort the value of Standardized Residuals  
qq\_df <- data.frame(residual\_std = sort(df$residual\_std))  
  
# add rank -> start with 1 for the smallest value  
qq\_df$rank <- c(1:n)  
  
# check percentile or quantile -> show the percentage of rank among overall  
qq\_df$quantile <- (qq\_df$rank - 0.5) / n  
  
# check qnorm of each quantile  
qq\_df$qnorm <- qnorm(qq\_df$quantile)  
  
qq\_df

## residual\_std rank quantile qnorm  
## 1 -1.36869452 1 0.03846154 -1.7688250  
## 2 -1.14355838 2 0.11538462 -1.1983797  
## 3 -0.86406116 3 0.19230769 -0.8694238  
## 4 -0.72775787 4 0.26923077 -0.6151411  
## 5 -0.61968092 5 0.34615385 -0.3957253  
## 6 -0.30513604 6 0.42307692 -0.1940281  
## 7 -0.11460998 7 0.50000000 0.0000000  
## 8 -0.05960895 8 0.57692308 0.1940281  
## 9 0.16379680 9 0.65384615 0.3957253  
## 10 0.97001543 10 0.73076923 0.6151411  
## 11 1.06118511 11 0.80769231 0.8694238  
## 12 1.25730849 12 0.88461538 1.1983797  
## 13 2.23205843 13 0.96153846 1.7688250

Plot the data to check normality

qq\_df %>%   
 ggplot(aes(x = qnorm, y = residual\_std)) +  
 geom\_point(size = 3) +  
 geom\_qq\_line(aes(sample = residual\_std), line.p = c(0.25, 0.75), size = 1, color = "magenta") +   
 labs(title = "Normality Plot for Standardized Residuals",  
 x = "Normal Score",   
 y = "Standardized Residuals")

