03/22/2020

Carlos Alberto Cordero Robles

Aplicación de filtrado colaborativo

Introducción

En esta tarea lo que se hará es usar una base de datos de compras que hicieron usuarios en la página de video juegos Steam [1]. La base de datos original cuenta con 200 mil compras/juego. Esta base de datos tuvo que ser modificada por fines prácticos o por el mismo funcionamiento del lenguaje R. Al final lo que se hace es hacer recomendaciones de compra a usuarios mediante la técnica de filtrado colaborativo y solo recomendando los video juegos que arrojaran una recomendación mayor al 80% para dicho usuario.

Dataset

El data set puede ser descargado directamente de internet [1]. Contiene una lista de usuarios y la acción que hacen sobre el video juego. El formato de la base de datos es un CSV y el orden de las columnas es el siguiente:

1 User ID

2 Juego

3 Accion (purchase/play)

4 value1

5 value2

Se tuvieron que hacer 3 modificaciones a la base de datos original para esta tarea en particular. Primero se redujo la base de datos de 200 mil a solo 20 mil muestras, de lo contrario el estudio tomaría más de 8 horas en correr. El segundo cambio agregar el título a cada columna, la base de datos original no lo tiene y la función read.csv toma la primera fila como si fueran los títulos. Por último, se remplazaron todos los espacios por puntos. Esto debido a que al parecer la función data.frame agrega puntos donde ve espacios por lo tanto para mantener consistencia a lo largo de todo el código con respecto a los nombres de los video juegos me fue muy práctico manejarlo con puntos en vez de espacios. Después de procesar la información la matriz con la que se trabajó contendrá 1235 usuarios y 2529 video juegos.

Proceso

A diferencia de la base de datos trabajada en clase, esta base de datos no está el mapeo de todos los usuarios junto con sus preferencias actuales. Debido a que el formato es diferente, la primera acción es crear una matriz con el formato usuario vs video juego donde se pondrá un 1 si ya compro el juego y 0 de lo contrario. Como se mencionó, este sistema de recomendaciones es sobre las compras, entonces las muestras que tengan la acción igual a play serán ignorados.

|  |
| --- |
| # Create Dataset  # Get users  for(user in music.usage[,"user"]){  if(FALSE ==user %in% userNames){  userNames[userGameIndex] <- user  userGameIndex <- userGameIndex + 1  }  }  userGameIndex <- 1  # Get Games  for(game in music.usage[,"game"]){  if(FALSE == game %in% gamesNames){  gamesNames[userGameIndex] <- game  userGameIndex <- userGameIndex + 1  }  }  games.usg.dist <- matrix(0, nrow=length(userNames),  ncol=length(gamesNames),  dimname=list(userNames,  gamesNames))  dim(games.usg.dist)  # We want to suggest to purchase not to play  for (actionIndex in 1:length(music.usage[,"action"])){  if("purchase"== music.usage[actionIndex,"action"]){  games.usg.dist[toString(music.usage[actionIndex,"user"]),music.usage[actionIndex,"game"]] <- 1  }  } |

Después se calcula la distancia de cada uno de los vectores o columna de video juegos.

|  |
| --- |
| # Now we have to calculate this distance(cos(angle)) between every video game vector  music.usg.dist <- matrix(NA, nrow=ncol(games.usg.dist),  ncol=ncol(games.usg.dist),  dimname=list(colnames(games.usg.dist),  colnames(games.usg.dist)))  for (i in 1:ncol(games.usg.dist)) {  print(gamesNames[i])  print(i)  for(j in 1:ncol(games.usg.dist)) {  music.usg.dist[i,j] <- dist.cosine(games.usg.dist[,i],  games.usg.dist[,j])  }  } |

Por último, crearemos una matriz user vs video juego donde la calificación será -1 si ya ha comprado el video juego y tendera a ser 1 conforme más se recomiende al usuario comprar dicho juego.

NOTA IMPORTANTE: Por alguna razón al convertir a data frame la matriz para poder trabajarla (music.usage.dist.df <- data.frame(music.usg.dist)) algunas columnas cambian su nombre y ya no pueden ser identificadas en el procesamiento, por el momento a todos esos video juegos les tuve que asignar el valor de -1. Fueron alrededor de 500 video juegos los que quedaron ignorados por esta situación, de cualquier modo, quedaron cerca de 2000 para el análisis. Queda para investigación saber porque al convertir a data frame el nombre de algunas columnas cambia.

|  |
| --- |
| for (i in 1:nrow(user.data)) {  for (j in 1:ncol(user.data)){  user <- rownames(user.data)[i]  artist <- colnames(user.data)[j]  # validaci??n 1  if(FALSE == (artist %in% colnames(music.usage.dist.df))){  #print(artist)  user.data[user,artist] = -1;  }  else  {  if (games.usg.dist[user,artist]==1) {  user.data[user,artist] = -1;  }  else  {  # Take the value of the cosine distance of the 6 more similar artist to the provided artist  top.artists <- head(n=6,  music.usage.dist.df[order(  music.usage.dist.df[,artist], decreasing=T),][artist])  # Remove the first one because it is the same artist that the provided  top.artists.names <- rownames(top.artists)[-1]  top.artists.sim <- (top.artists[,1])[-1]    # Take the answer (1 or 0) from the user for the calculated 5 more similar groups to the group analized  top.artists.history <-  games.usg.dist[user,top.artists.names]  # give a score for such artist for the current user  user.data[user,artist] <-  score.recommendation(top.artists.history, top.artists.sim)  }  }  }  } |

Para obtener las recomendaciones hago un barrido sobre los resultados e imprimo los que dieran resultado mayor al 80%.

|  |
| --- |
| #Recomendations over 0.8  for (i in 1:nrow(user.data)) {  for (j in 1:ncol(user.data)){  user <- rownames(user.data)[i]  artist <- colnames(user.data)[j]  if(is.na(user.data[i,j]) == FALSE){  if(user.data[i,j] > 0.8){  print(c("We suggest to user: ", user, "the game: ", artist))  }  }  }  } |

Análisis de resultados

Algunas de las recomendaciones que proporciono el algoritmo fueron las siguientes.

|  |
| --- |
| "We suggest to user: " "151603712""the game: ""Microsoft.Flight.Simulator.X.Steam.Edition"  "We suggest to user: " "53875128" "the game: " "Blocks.That.Matter"  "We suggest to user: " "53875128" "the game: " "Super.Killer.Hornet.Resurrection"  "We suggest to user: " "53875128" "the game: " "Fortix"  "We suggest to user: " "53875128" "the game: " "Iron.Grip.Warlord"  "We suggest to user: " "53875128" "the game: " "Assetto.Corsa"  "We suggest to user: ""53875128""the game: ""Front.Mission.Evolved"  "We suggest to user: " "126340495""the game: " "Wizardry.Online"  "We suggest to user: " "226212066""the game: " "Games.of.Glory"  "We suggest to user: " "65117175" "the game: " "LIMBO"  "We suggest to user: " "65117175" "the game: " "Cortex.Command"  "We suggest to user: ""11373749""the game: ""Hitman.Sniper.Challenge"  "We suggest to user: " "11373749" "the game: " "Rochard"  "We suggest to user: " "11373749" "the game: " "Day.of.Defeat.Source"  "We suggest to user: ""11373749""the game: ""The.Walking.Dead.Season.Two"  "We suggest to user: " "11373749" "the game: " "Startopia"  "We suggest to user: ""11373749""the game: ""Magicka.Holiday.Spirit.Item.Pack."  "We suggest to user: " "11373749" "the game: " "Fortix"  "We suggest to user: " "11373749" "the game: " "Iron.Grip.Warlord"  "We suggest to user: ""11373749""the game: ""Call.of.Juarez.The.Cartel"  "We suggest to user: ""11373749""the game: ""Front.Mission.Evolved"  "We suggest to user: " "54103616" "the game: " "Day.of.Defeat.Source"  "We suggest to user: " "54103616" "the game: " "Train.Simulator"  "We suggest to user: " "298389371""the game: " "Train.Simulator" |

Tomemos por ejemplo la sugerencia "We suggest to user: ""11373749""the game: ""Hitman.Sniper.Challenge".

Primero veremos que el usuario no cuenta actualmente con dicho juego.

|  |
| --- |
| games.usg.dist["11373749","Hitman.Sniper.Challenge"]  0 |

Los video juegos top más cercanos a el videojuego Hitman.Sniper.Challenge son los siguientes con sus respectivas distancias donde 1 es la menor distancia y -1 es la mayor.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| "Hitman.Absolution" | "Hitman.Codename.47" | "Hitman.Blood.Money" | "Hitman.Contracts" | "Hitman.2.Silent.Assassin" |
| 0.8164966 | 0.5533986 | 0.5276449 | 0.5103104 | 0.5051815 |

Luego veremos que de esos juegos solo Hitman.Contracts no lo tiene el usuario.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| "Hitman.Absolution" | "Hitman.Codename.47" | "Hitman.Blood.Money" | "Hitman.Contracts" | "Hitman.2.Silent.Assassin" |
| 1 | 1 | 1 | 0 | 1 |

La recomendación entonces da de **0.8248181,** por lo tanto, si es muy altamente recomendable para el usuario 11373749 la adquisision del juego Hitman Sniper Challenge lo cual hace sentido porque él ya cuenta con varios de los juegos de Hitman.

Código

|  |
| --- |
| # Read the data  getwd()  setwd("C:/Users/ccordero/Desktop/Maestria/09\_MAIM/03\_filtrado/steam-video-games/")  music.usage <- read.csv('steam-20k.csv', stringsAsFactors = F)  gamesNames <- vector()  userNames <- vector()  userGameIndex <- 1  # Create Dataset  # Get users  for(user in music.usage[,"user"]){  if(FALSE ==user %in% userNames){  userNames[userGameIndex] <- user  userGameIndex <- userGameIndex + 1  }  }  userGameIndex <- 1  # Get Games  for(game in music.usage[,"game"]){  if(FALSE == game %in% gamesNames){  gamesNames[userGameIndex] <- game  userGameIndex <- userGameIndex + 1  }  }  games.usg.dist <- matrix(0, nrow=length(userNames),  ncol=length(gamesNames),  dimname=list(userNames,  gamesNames))  dim(games.usg.dist)  # We want to suggest to purchase not to play  for (actionIndex in 1:length(music.usage[,"action"])){  if("purchase"== music.usage[actionIndex,"action"]){  games.usg.dist[toString(music.usage[actionIndex,"user"]),music.usage[actionIndex,"game"]] <- 1  }  }  games.usg.dist["151603712","The Elder Scrolls V Skyrim"]  games.usg.dist["187131847","The Elder Scrolls V Skyrim"]  games.usg.dist["187131847","Dota 2"]  games.usg.dist["128470551","RUSH"]  # We need to check that all that data is consistent TRUE  complete.cases(music.usage)  # Display data  str(music.usage)  # Check the first elemets  head(music.usage)  # User index  rownames(music.usage)  # colums  colnames(music.usage)  # preferences of user 1, check how the information is displayed, the columns names is also displayed.  music.usage[1,]  # We can check one column using $  music.usage$game  length(music.usage$game)  # Remove the user column to have just valuable information  #music.usage.nouser <- music.usage[,2:ncol(music.usage)]  #music.usage.nouser <- music.usage[,-1]  #music.usage.nouser <- music.usage  #music.usage.nouser$user <- NULL  #head(music.usage.nouser)  #class(music.usage.nouser)  # Function to get the cosin of the angle between vectors  vector.1 <- c(1,2,3)  magnit.vec <- function(v1) {  return(sqrt(sum(v1 \* v1)))  }  prod.punt <- function(v1,v2) {  return(sum(v1 \* v2))  }  dist.cosine <- function(v1, v2) {  return(prod.punt(v1,v2)/(magnit.vec(v1)\*magnit.vec(v2)))  }  magnit.vec(vector.1)  # How parallel are they, 1 is the maximum  dist.cosine(c(1,1,1),c(1,1,1))  dist.cosine(c(1,0,1),c(0,1,0))  dist.cosine(c(5,10,100),c(-5,-10,-99))  dist.cosine(c(1,1,1,1),c(1,1,1,1000))  dist.cosine(c(1,1,1,1),c(3,3,3,3))  # Now we have to calculate this distance(cos(angle)) between every video game vector  music.usg.dist <- matrix(NA, nrow=ncol(games.usg.dist),  ncol=ncol(games.usg.dist),  dimname=list(colnames(games.usg.dist),  colnames(games.usg.dist)))  for (i in 1:ncol(games.usg.dist)) {  print(gamesNames[i])  print(i)  for(j in 1:ncol(games.usg.dist)) {  music.usg.dist[i,j] <- dist.cosine(games.usg.dist[,i],  games.usg.dist[,j])  }  }  dim(music.usg.dist)  # Lets display some results  #colnames(games.usg.dist)  music.usg.dist[1:5,1:5]  # Now lets find the 10 users more similar to a provided user  # We need first to convert again to data frame the matrix created (as matrix it can not be computed)  music.usage.dist.df <- data.frame(music.usg.dist)  music.usage.similar.artists <- matrix(NA, nrow=ncol(music.usage.dist.df),  ncol=10,  dimname=list(gamesNames))  dim(music.usage.similar.artists)  # we can agroup by similarity if we just order the vectors  rownames(music.usage.dist.df[order(music.usage.dist.df[,1], decreasing=T),])[1:10]  # Lets apply this logic but for all collumns the first one is allways the group itself then we are going to take from 2 to 11  for(i in 1:ncol(music.usage.dist.df)) {  music.usage.similar.artists[i,] <-  rownames(  music.usage.dist.df[  order(music.usage.dist.df[,i], decreasing=T),])[2:11]  }  music.usage.similar.artists["The.Elder.Scrolls.V.Skyrim",]  music.usage.similar.artists["Fallout.4",]  # Now we are going to provide recomendations  # Lets map User vs group  user.data <- matrix(NA, nrow=length(userNames),  ncol=length(gamesNames),  dimname=list(userNames,  gamesNames))  dim(user.data)  # The score of one artist will consider the cosine distance of the artist that like the user (the value is normalized using all the cosine distances)  # For example if history is all 1s the the recomendation score will be 1.0  # Apparently this function works better if our data has many 0s  score.recommendation <- function(history, similarities) {  return(sum(history\*similarities)/sum(similarities))  }  # Apparently this function works better if our data has many 1s  #score.recommendation <- function(history, similarities) {  # return(sum(history\*similarities)/length(history))  #}  # Clean up  # if(FALSE == ("The.Elder.Scrolls.V.Skyrim" %in% colnames(music.usg.dist))){  # print("Dragon.Age.Origins.-.Ultimate.Edition")  # }  #  # if(FALSE == ("The.Elder.Scrolls.V.Skyrim" %in% colnames(music.usage.dist.df))){  # print("Dragon.Age.Origins.-.Ultimate.Edition")  # }  #  # i <- 2  # for (j in 1:ncol(user.data)){  # user <- rownames(user.data)[i]  # artist <- colnames(user.data)[j]  # if(FALSE == (artist %in% colnames(music.usage.dist.df))){  # #print(artist)  # user.data[user,artist] = -1;  # }  # else  # {  # if (games.usg.dist[user,artist]==1) {  # user.data[user,artist] = -1;  # }  # else  # {  # # Take the value of the cosine distance of the 6 more similar artist to the provided artist  # top.artists <- head(n=6,  # music.usage.dist.df[order(  # music.usage.dist.df[,artist], decreasing=T),][artist])  # # Remove the first one because it is the same artist that the provided  # top.artists.names <- rownames(top.artists)[-1]  # top.artists.sim <- (top.artists[,1])[-1]  #  # # Take the answer (1 or 0) from the user for the calculated 5 more similar groups to the group analized  # top.artists.history <-  # games.usg.dist[user,top.artists.names]  # # give a score for such artist for the current user  # user.data[user,artist] <-  # score.recommendation(top.artists.history, top.artists.sim)  # }  # }  # }  # user.data[user,]  # Clean up  for (i in 1:nrow(user.data)) {  for (j in 1:ncol(user.data)){  user <- rownames(user.data)[i]  artist <- colnames(user.data)[j]  # validaci??n 1  if(FALSE == (artist %in% colnames(music.usage.dist.df))){  #print(artist)  user.data[user,artist] = -1;  }  else  {  if (games.usg.dist[user,artist]==1) {  user.data[user,artist] = -1;  }  else  {  # Take the value of the cosine distance of the 6 more similar artist to the provided artist  top.artists <- head(n=6,  music.usage.dist.df[order(  music.usage.dist.df[,artist], decreasing=T),][artist])  # Remove the first one because it is the same artist that the provided  top.artists.names <- rownames(top.artists)[-1]  top.artists.sim <- (top.artists[,1])[-1]    # Take the answer (1 or 0) from the user for the calculated 5 more similar groups to the group analized  top.artists.history <-  games.usg.dist[user,top.artists.names]  # give a score for such artist for the current user  user.data[user,artist] <-  score.recommendation(top.artists.history, top.artists.sim)  }  }  }  }  head(user.data)  # Ignore the error, maybe there is a null in the matrix  # Recomendation for user 1  head(sort(user.data[4,], decreasing = T))  #Recomendations over 0.8  for (i in 1:nrow(user.data)) {  for (j in 1:ncol(user.data)){  user <- rownames(user.data)[i]  artist <- colnames(user.data)[j]  if(is.na(user.data[i,j]) == FALSE){  if(user.data[i,j] > 0.8){  print(c("We suggest to user: ", user, "the game: ", artist))  }  }  }  }  # Lets analyze the result user 11373749, artist limp.bizkit  games.usg.dist["11373749","Hitman.Sniper.Challenge"] # is not liked before = 0  games.usg.dist["11373749","Hitman.Sniper.Challenge"]  top.artists <- head(n=6,  music.usage.dist.df[order(  music.usage.dist.df[,"Hitman.Sniper.Challenge"], decreasing=T),]["Hitman.Sniper.Challenge"])  top.artists.names <- rownames(top.artists)[-1]  top.artists.sim <- (top.artists[,1])[-1]  top.artists.names  top.artists.sim  top.artists.history <-  games.usg.dist["11373749",top.artists.names]  # linkin.park papa.roach rammstein disturbed system.of.a.down  #11373749 1 1 1 1 0  user.hitman.recomendation <- score.recommendation(top.artists.history, top.artists.sim)  # 0.8286139 Highly recomended |

Conclusiones

\* El filtrado colaborativo es una manera muy útil para encontrar recomendaciones a usuarios.

\* Es necesario una base de datos de los usuarios contra las distintas opciones para poder hacer una recomendación.

\* La recomendación estará limitada las opciones seleccionadas por los usuarios en la base de datos.

\* Mientras más grande sea la base de datos más robusta es la recomendación.

\* Mientras más grande sea la base de datos el procesamiento será mayor y tardara más en ejecutarse.

\* Los resultados de las sugerencias por filtrado colaborativo son bastante congruentes.

Referencias

[1] « Steam Video Games». [En línea]. Disponible en:

<https://www.kaggle.com/tamber/steam-video-games/data#> [Accedido: 23-Mar-2020].