**Tecnológico Nacional de México**

**Instituto Tecnológico de Tijuana**

**Subdirección Académica**

**Departamento de Sistemas y Computación**

**SEMESTRE:**

Febrero-Julio 2021

**CARRERA:**

Ingeniería en Sistemas Computacionales e Ingeniería en Tecnologías de la Información y Comunicaciones

**MATERIA Y SERIE:**

Minería de Datos

BDD-1703 TI9A

**UNIDAD A EVALUAR:**

Unidad III

**NOMBRE DEL TRABAJO:**

**Práctica #2**

Documentación del programa K-NN en R

**NOMBRE Y NÚMERO DE CONTROL DE LOS INTEGRANTES:**

Rodriguez Medrano Marco Antonio 17210635

**NOMBRE DEL DOCENTE:**

José Christian Romero Hernández

**Tijuana, Baja California Jueves 30 de Mayo de 2021**

Instrucciones: se le pidió al alumno que documentara y describa el siguiente código en R, y desplegará los resultados

Con el comando getwd obtenemos el directorio del repositorio donde estamos trabajando

|  |
| --- |
| *# K-Nearest Neighbors (K-NN)* getwd() setwd("/Users/DELL/Desktop/DataMining/MachineLearning/knn") getwd()  Una vez obtenida la ubicación de nuestro repositorio importamos el dataset que utilizaremos en esta ocasión y escogeremos únicamente la columna 3 y 5. *# Importing the dataset* dataset = read.csv('Social\_Network\_Ads.csv') dataset = dataset[3:5]  Ahora vamos a codificar como factor de 0 y 1 la columna purchased, para trabajar más adelante con ella. *# Encoding the target feature as factor* dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))  *# Splitting the dataset into the Training set and Test set* *# install.packages('caTools')*  Aquí en estas 5 líneas haremos la separación del dataset utilizando la librería caTools y un split simple con un radio del 75%, esto hará que de los 400 datos 100 sean de y los 300 restantes de entrenamiento. library(caTools) set.seed(123) split = sample.split(dataset$Purchased, SplitRatio = 0.75) training\_set = subset(dataset, split == TRUE) test\_set = subset(dataset, split == FALSE)  Ahora hacemos al set de entrenamiento y prueba sus escalas que serán utilizadas para obtener su predicción.  *# Feature Scaling* training\_set[-3] = scale(training\_set[-3]) test\_set[-3] = scale(test\_set[-3])  Utilizando las escalas ahora si hacemos la predicción de los set de prueba y entrenamiento un un valor de k de 5.  *# Fitting K-NN to the Training set and Predicting the Test set results* library(class) y\_pred = knn(train = training\_set[, -3],  test = test\_set[, -3],  cl = training\_set[, 3],  k = 5,  prob = TRUE)  Creamos la matriz de confusión para hacer la clasificación de los datos.  *# Making the Confusion Matrix* cm = table(test\_set[, 3], y\_pred)  Aquí vamos a visualizar los resultados del set de entrenamiento esto utilizando la librería ElemStatLearn la cual tiene un libro y si lo quieren leer pueden buscarlo en esta liga:[https://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII\_print12\_toc.pd](https://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII_print12_toc.pdf), despues guardaremos el set de entrenamiento en una variable llamada set y después crearemos otras dos variables llamadas X1 y X2 que nos ayudarán a graficar los resultados y dichas variables las expandimos con el comando expand.grid y lo guardamos en la variable grid\_set al mismo que cambiaremos el nombre de las columnas por otros más adecuados.  *# Visualising the Training set results* library(ElemStatLearn) set = training\_set X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01) X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01) grid\_set = expand.grid(X1, X2) colnames(grid\_set) = c('Age', 'EstimatedSalary')  En esta línea calculamos a knn utilizando el set de entrenamiento al que posteriormente haremos el ploteo o graficación. y\_grid = knn(train = training\_set[, -3], test = grid\_set, cl = training\_set[, 3], k = 5) plot(set[, -3],  main = 'K-NN (Training set)',  xlab = 'Age', ylab = 'Estimated Salary',  xlim = range(X1), ylim = range(X2))  resultado    Se agrega la matriz para hacer la división de los datos y de la región.  contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)  Resultado    Pero como observamos no podemos deducir nada con la anterior gráfica para ello le daremos color con los siguientes comandos.  points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato')) points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))  Resultado  Aquí simplemente repetimos los mismos pasos pero solo con el cambio de set que utilizaras el cual ahora será el set de prueba.  *# Visualising the Test set results* library(ElemStatLearn) set = test\_set X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01) X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01) grid\_set = expand.grid(X1, X2) colnames(grid\_set) = c('Age', 'EstimatedSalary') y\_grid = knn(train = test\_set[, -3], test = grid\_set, cl = test\_set[, 3], k = 5) plot(set[, -3],  main = 'K-NN (Test set)',  xlab = 'Age', ylab = 'Estimated Salary',  xlim = range(X1), ylim = range(X2)) contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE) points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato')) points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))  Resultados |