**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 #4**

Documentación del programa random forest 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 Martes 01 de Junio de 2021**

Instrucciones: se le pidió al alumno que documente la visualización de los datos del programa random forest en R.

Antes de hablar de la visualización se trabajaron los datos, primeramente cargándola después hicimos la separación de los datos en dos set’s set de entrenamiento y de prueba a los cuales les creamos escalas para interpretar de mejor manera los datos. Una vez hecho esto lo siguiente será crear la clasificación random forest, la predicción y por último la matriz de confusión.

Para la visualización como lo hemos trabajado en otras prácticas cargamos la librería ElemStatLearn la cual tiene su origen de un libro que lleva su mismo nombre pueden buscar en google el libro para mayor información una vez cargada la librería inicializamos con la variable set los datos con los que trabajaremos los cuales son los de entrenamiento, de ahí creamos dos variables llamadas X1 y X2 las cuales nos serviran mas adelante para hacer los rangos de los límites de la coordenada “x”, de ahí con el comando expand.drid expandimos las variables X1 y X2 despues cambiamos el nombre de las columnas de los datos.

Después creamos una variable llamada y\_grid a la cual le cargaremos la predicción que hicimos antes de la visualización esto para después ser utilizada para hacer el ploteo(grafica) utilizando las variables X1 y X2 como límites.

|  |
| --- |
| *# 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') y\_grid = predict(classifier, grid\_set) plot(set[, -3],  main = 'Random Forest Classification (Training set)',  xlab = 'Age', ylab = 'Estimated Salary',  xlim = range(X1), ylim = range(X2))    Pero como podemos observar an no nos dice nada la gráfica, para ello utilizamos el comando contour, el cual le agrega un contorno el cual se dará por el valor de la matriz de confusión  contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = TRUE)    Bueno aun no nos dice mucho con este contorno se nos está dando a entender que hay dos regiones con sus respectivos puntos que todavía no sabemos cuales son y donde pertenecen, para ello utilizamos los siguientes dos comandos, con el primero obtendremos las regiones. points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'springgreen3', 'tomato'))    Pero aún nos faltan los puntos, bueno para ello utilizamos el segundo comando.  points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))  Para la otra visualización es básicamente seguir los mismos pasos pero con el único cambio de que ahora los datos que se van a visualizar serán los 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 = predict(classifier, grid\_set) plot(set[, -3], main = 'Random Forest Classification (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          Este último comando podremos ver la estimación de error en los diferentes árboles del clasificador. *# Choosing the number of trees* plot(classifier) |