EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO®

Tecnológico Nacional de México
Instituto Tecnológico de Tijuana

Subdirección Académica
Departamento de Sistemas y Computación
Ingeniería en Sistemas Computacionales
Semestre: AGOSTO-DICIEMBRE 2021

MINERÍA DE DATOS

BDD-1703SC9A

"Practice 4"

Jiménez Ramírez Julio Fabián	17212147
Flores González Luis Diego	C16211486

MC. JOSE CHRISTIAN ROMERO HERNANDEZ

Campus Tomas Aquino

"Por una juventud integrada al desarrollo de México"

Tijuana B.C. a 29 de Noviembre del 2021

Practice 4

In this practice, an example of the KNN model will be developed, firstly, the data will be stored in a variable using the "Read" method, to have a better use of the data it is necessary to select only the fields' Age, estimated salary and purchased " .

```
# Importing the dataset
dataset = read.csv('Social_Network_Ads.csv')
dataset = dataset[3:5]
```

The "factor" function is used to encode a vector as a factor (the terms "category" and "enumerated type" are also used for factors), we change the categorical values to 0 and 1.

```
# Encoding the target feature as factor
dataset$Purchased = factor(dataset$Purchased, levels = c(0, 1))
```

To perform the division of the dataset for training and testing we use the "caTools" library, specifying the randomness coefficient as 123. The data will be divided into a 75% margin for training and the remainder for testing, each percentage of dataset will have its own independent variable.

```
# Splitting the dataset into the Training set and Test set
# install.packages('caTools')
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)
```

We scale the training and test data to perform better on predictions.

```
# Feature Scaling
training_set[-3] = scale(training_set[-3])
test_set[-3] = scale(test_set[-3])
```

To make the prediction it is necessary to import the "class" library which contains the knn function where the values of:

- train: using the specific training data in the “Age” and “EstimatedSalary” fields.
- test: using the specific test data in the "Age" and "EstimatedSalary" fields.
- cl: using the "Purchased" field as the true classification factor.
- k: using 5 as the neighbor number of the training dataset.
- prob: using "true".

```
# 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)
y_pred
```

Result:

```
> y_pred
  [1] 0 0 0 0 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0
 [50] 0 0 0 1 1 1 0 1 0 0 1 0 0 0 1 1 0 1 1 1 1 1 1 1 0 0 0 1 0 0 1 0 1 0 1 0
1 1 0 0 1 1 0 1 0 1 1 1 1
 [99] 0 1
attr(,"prob")
  [1] 1.0 1.0 1.0 1.0 1.0 0.8 0.8 1.0 0.6 1.0 1.0 1.0 0.8 1.0 1.0 1.0 1.0 1.0
1.0 1.0 0.6 1.0 1.0 0.8
 [25] 1.0 0.8 1.0 1.0 0.8 1.0 1.0 0.8 0.8 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.8 1.0
1.0 1.0 0.8 1.0 1.0 1.0
 [49] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.8 0.8 0.8 1.0 0.6 1.0 0.6 1.0
0.8 0.8 1.0 0.8 0.8 0.8
 [73] 0.8 0.8 0.6 1.0 0.8 1.0 1.0 1.0 1.0 0.8 1.0 1.0 0.8 0.8 0.8 0.6 0.8 1.0
0.8 0.8 0.8 0.8 1.0 0.6
 [97] 1.0 0.8 1.0 1.0
Levels: 0 1
```

To verify the correct use of the data, it shows a confusion matrix, using the test set and the prediction as parameters.

```
# Making the Confusion Matrix
cm = table(test_set[, 3], y_pred)
cm
```

Result:

```
> cm
   y_pred
     0  1
  0 59  5
  1  6 30
```

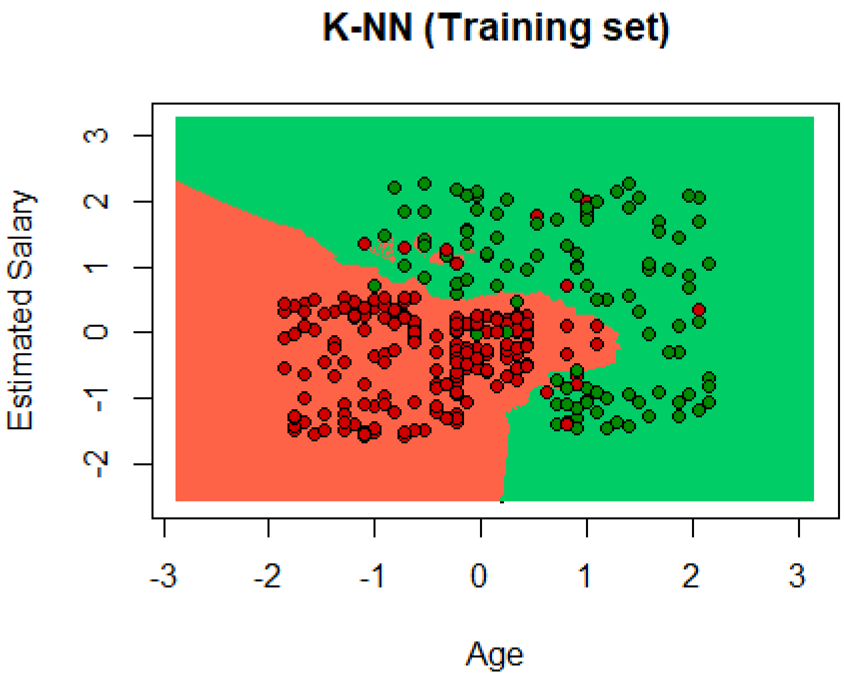
Finally we will show a graph using the "ElemStatLearn" library that allows a better visualization of the KNN model with the training data.

1. We import the library
2. In a new variable we define the data of the training dataset
3. We create 2 new variables with the transformation of the values of the “Age” field for “X1” and the “Estimated Salary” field for “X2”
4. We create a data frame "grid_set" from all the combinations of the vectors or factors provided previously
5. The names of the fields are changed to their previous ones
6. We generate a new variable "y_grid" for the prediction, where the method "KNN" is used using the values of the previous example as a parameter, in this case changing the test parameter for grid_set

7. We start the visualization of the data, defining the main name as well as the X axis and the Y axis, marking the limits of both with the values obtained from the variables X1 and X2
8. We add a contour line to the existing graph, using the "contour" method with the parameters of the sequences X1 and X2, as well as the values obtained in the prediction "y_grid"
9. Ending with the "points" methods that allow to make a change in the design of the graph to visually show the separation of the categories by sections and the points as such in different colors

```
# 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 = 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))
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'))
```

Result:



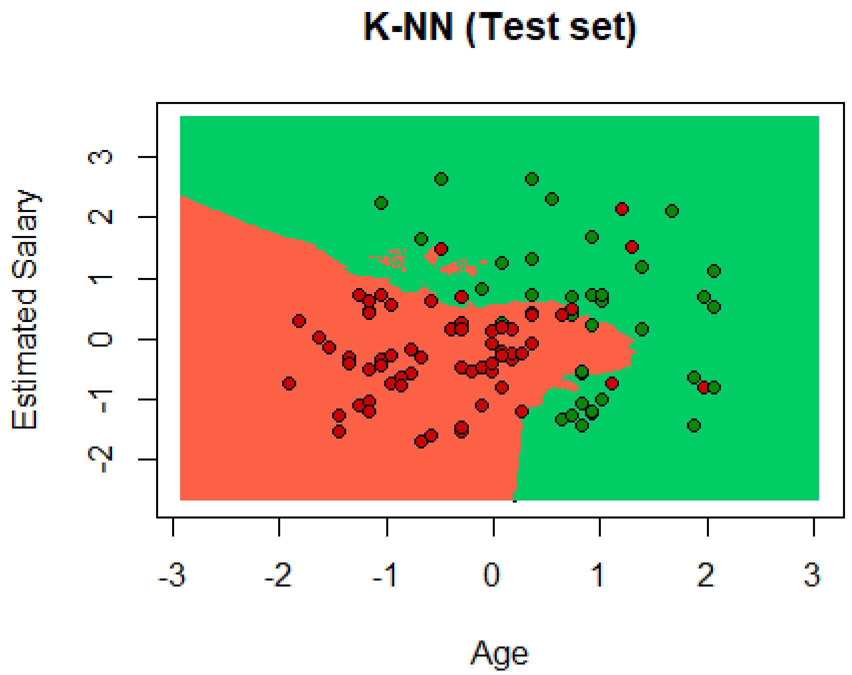
In the same way as the previous graph, now with the test dataset we use the "ElemStatLearn" library that allows a better visualization of the KNN model with the test data.

1. We import the library
2. In a new variable we define the data of the test dataset
3. We create 2 new variables with the transformation of the values of the “Age” field for “X1” and the “Estimated Salary” field for “X2”
4. We create a data frame "grid_set" from all the combinations of the vectors or factors provided previously
5. The names of the fields are changed to their previous ones
6. We generate a new variable "y_grid" for the prediction, where the method "KNN" is used using the values of the previous example as a parameter, in this case changing the test parameter for grid_set
7. We start the visualization of the data, defining the main name as well as the X axis and the Y axis, marking the limits of both with the values obtained from the variables X1 and X2
8. We add a contour line to the existing graph, using the "contour" method with the parameters of the sequences X1 and X2, as well as the values obtained in the prediction "y_grid"
9. Ending with the "points" methods that allow to make a change in the design of the graph to visually show the separation of the categories by sections and the points as such in different colors

```
# 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 = training_set[, -3], test = grid_set, cl = training_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'))
```

Result:


K-NN (Test set)
Estimated Salary
Age