Tecnológico Nacional de México
Instituto Tecnológico de Tijuana

Subdirección Académica
Departamento de Sistemas y Computación

Semestre:
Febrero – Junio 2021

Carrera:
Ingeniería en Tecnologías de la Información y Comunicaciones

Materia y serie:
Minería de datos BDD-1703TI9A

Unidad a evaluar: Unidad II

Nombre de la Tarea:
Práctica 1

Nombre del Alumno:
Hernández Negrete Juan Carlos 16212021
Sifuentes Martinez Manuel Javier 17212934

Nombre del docente:
José Christian Romero Hernández

Analysis of data visualization in the logistic regression model

Change of directory

The functions that are used have been used throughout this semester to later load the dataset into R Studio's memory, using getwd to obtain the current path and setwd to assign a new path.

```
getwd ()
setwd ("../Desktop/DataMining/MachineLearning/LogisticRegression")
getwd ()
```

Import and section dataset

Once the directory has changed, through the read.csv function the dataset is loaded into memory. Reviewing the data, the decision is made that only the last 3 columns are necessary to make the analysis that will be carried out, so all the rows are selected, but in the second parameter it is specified that only the Age columns will be taken , EstimatedSalary and Purchased.

```
dataset <- read.csv ('Social_Network_Ads.csv')
dataset <- dataset [, 3:5]
```

Divide the dataset into training and test sets

Loading the caTools library helps us to access its functions, which are essential to perform the present analysis. Set.seed is used to generate a sequence of "random" numbers, and a specific number is defined so that the results can be replicated. Then, thanks to caTools, sample.split is used to divide the dataset into 2, the training set, which will serve as its name says, to see how the algorithm behaves, and then the dataset with which it should deliver the expected results. sample.split takes as arguments the column of the table, and the percentage chosen for the number of data. At the end, through subset, the fields that have TRUE for the training set are assigned, and FALSE, which would be 25% of the resulting data that was not chosen from the sample.split function for the data set test.

```
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)
```

Scale of characteristics

In the graph, the Age and EstimatedSalary columns present very varied values, so when analyzing them the results may not be very exact. To minimize the margin of error, the scale function is used, as its name says, through this a scale will be generated with respect to the values that these columns have, giving as parameters the number of rows to be taken, the which in this case are all of them, and the columns, which are specified to be only 1 and 2. In the end, it is only equated to their respective dataset so that it is overwritten and takes the new values.

```
training_set [, 1:2] <- scale (training_set [, 1:2])
test_set [, 1:2] <- scale (test_set [, 1:2])
```

Adaptation of logistic regression to the training set

Enter now to what is the logistic regression with the help of the glm function, this to fit the model to the data. 3 parameters are shown, the first is the dependent variable, which corresponds to the Purchased column, since it is the result that we want to predict, so it cannot be changed. Then there are the independent variables (family is used to choose the type of distribution, in this practice the binomial is chosen, which is used for proportions, and then data is found, which is where the model variables are taken, which would be our training dataset), these are manipulated in order to see their effect and arrive at the expected result of the dependent variable. There are two signs "~" and ".", The first indicates that after it are the independent variables, and the second indicates that all the remaining variables of the dataset are used.

```
classifier = glm (formula = Purchased ~.,
                  family = binomial,
                  data = training_set)
```

Predicting the results of the test set

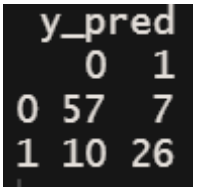
Through the predict function we will try to predict the results based on the data found in the classifier variable, the parameter type with "response" causes the result to be displayed as numeric, unlike using "class" which shows the label assigned to that value. Finally there is newdata, here we look for variables to predict, so column 3 of the dataset is given. The result that it returns is a number with several decimal places, so to make it more understandable the ifelse function is used, in which it is specified that if the value is greater than .5 it is set as 1, otherwise it is defined as 0. And in this way, you can compare whether or not you were successful with the results.

```
prob_pred = predict (classifier, type = 'response', newdata = test_set
[-3])
prob_pred
y_pred = ifelse (prob_pred> 0.5, 1, 0)
y_pred
```

Making a confusionconfusion
matrix Thematrix is used to evaluate the performance of a classification model, so, through this, the percentage of error and effectiveness of the created model will be seen. This has 4 options: true positives, false positives, false negatives and true negatives. Once this is explained, the table function is used to create said matrix, giving the test dataset as parameters, but only column 3, which is the one that corresponds to whether or not a purchase was made, and then there is the prediction made. previously.

```
cm = table (test_set [, 3], y_pred)returned
cm
```

The result that is the matrix shown below, in this it is observed that there is a 17% error in the prediction made, this since 10 and 7 correspond to false positive and negative.

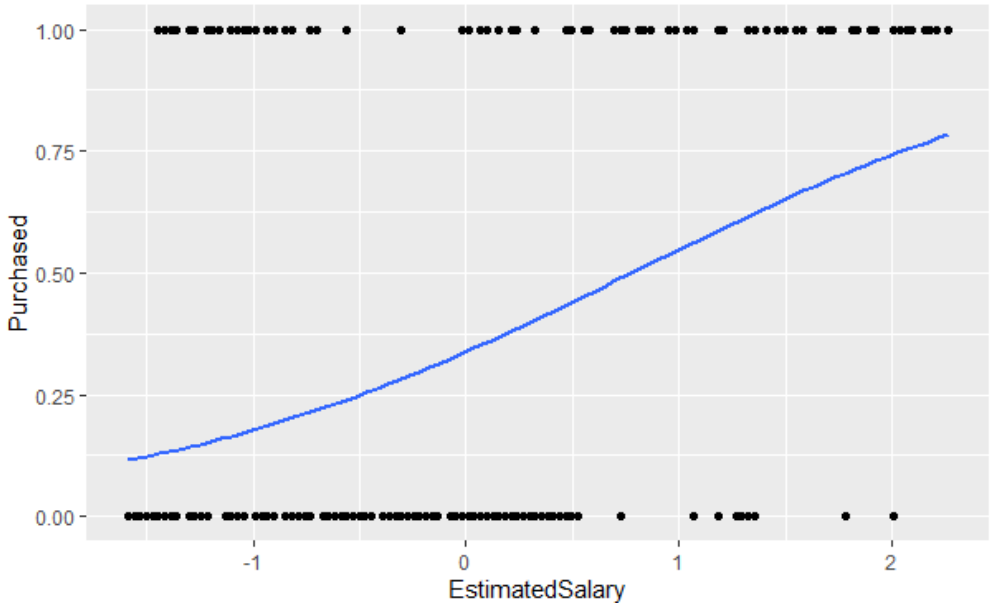


	y_pred	
	0	1
0	57	7
1	10	26

Graphical representation of results
The ggplot2 library is loaded to access the functions that allow it to construct the graphs. The construction of the graphs is done in the same way, only the variables equal to "x" and "y" and the dataset change. The first parameter corresponds to the source of the data, in this case the data found in the training set will be graphed, with the help of the aesthetics layer the “x” and “y” axes are defined. The geom_point function is used to show a distribution of points, since until now only the axes had been constructed. The stat_smooth function helps to see patterns. Two parameters were used, the first is method, it is a smoothing method, "glm" was used since it is the type of function that was used previously. method.args is used to create a list of additional arguments, this means that the graph is constructed with respect to a binomial distribution. The confidence interval around the formed line is shown, it is a shaded area that expands or moves depending on the entered results.

```
library(ggplot2)
ggplot (training_set, aes (x = EstimatedSalary, y = Purchased)) +
geom_point () + stat_smooth (method ="GLM"method.args = list (family
="binomial"),is =FALSE)
```

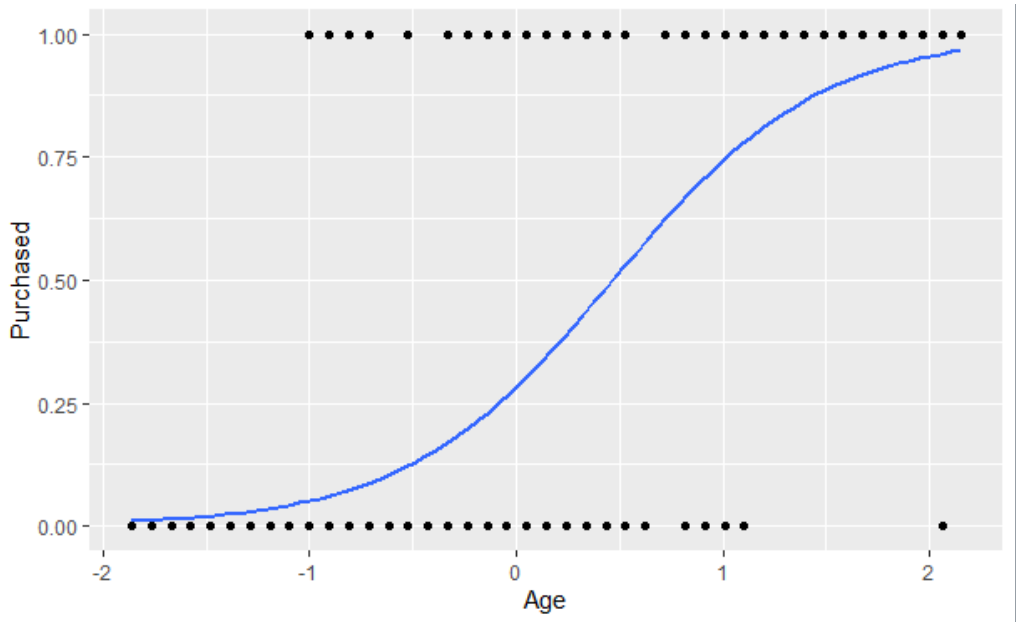
The Previous instructions result in this graph, the relationship between the estimated salary and the possibility of purchase is shown. A relationship close to 1: 1 is observed, there is not a very sharp curve, so there is no such thing as a binomial distribution.



Graph 1

It has the same parameters as the previous graph, except that the "x" axis equals age. In this way, it is seen that a graph is created, with an evident curve, resulting in the conclusion that it is a binomial distribution.

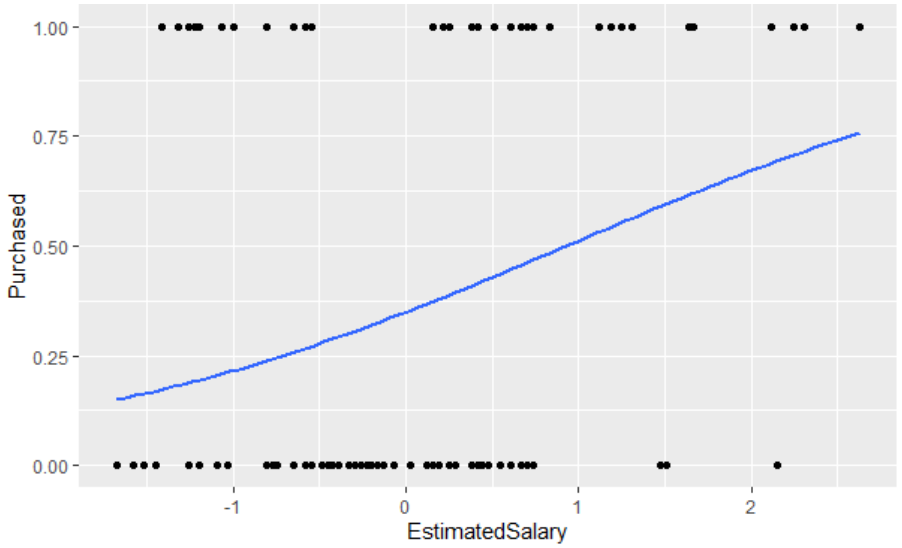
```
ggplot (training_set, aes (x = Age, y = Purchased)) + geom_point () +
  stat_smooth (method ="glm", method.args = list (family ="binomial"),
se =FALSE)
```



Graph 2

In this graph the data source is changed, now using the test dataset, but with the same "x" and "y" axes of the first graph. The result is again clear that it is not part of a normal distribution, as itcloser to being a straight line,

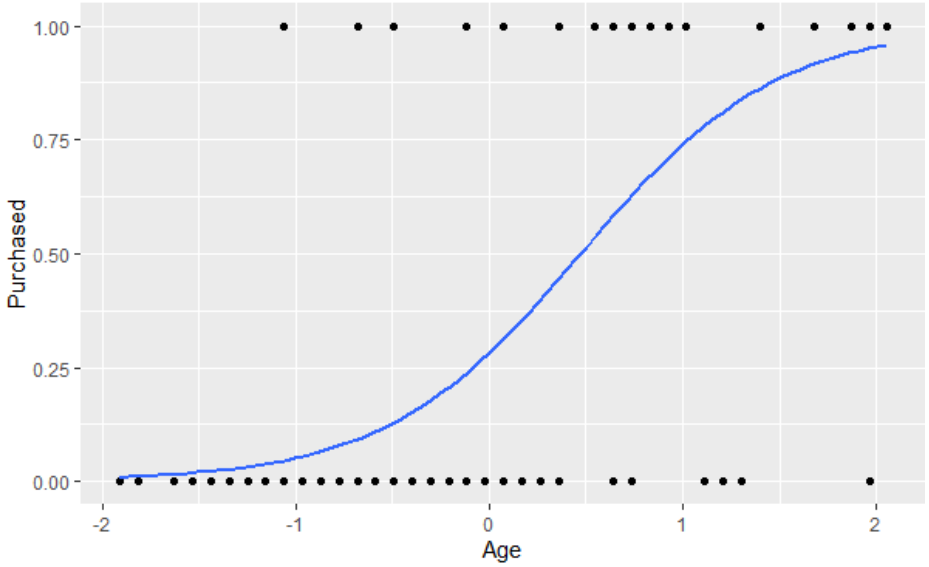
```
getsggplot (test_set, aes (x = EstimatedSalary, y = Purchased)) +
geom_point () +
  stat_smooth (method =" glm ", method.args = list (family =" binomial
"), se =FALSE)
```



Graph 3

In this graph, only the" x "axis changes when it is changed to the age field, and a clear binomial distribution can be visualized, having a marked curve in the age-purchase relationship.

```
ggplot (test_set, aes (x = Age, y = Purchased)) + geom_point () +
  stat_smooth (method ="glm", method.args = list (family ="binomial"),
se =FALSE)
```



Graph 4

Adaptation of the logistic regression to the training set

The following code tries to analyze the existing trends in the data set. The ElemStatLearn library is discontinued, but it is very useful, it was created for the book "The Elements of Statistical Learning", it contains many popular data sets.

First, the set is declared as a training set, saving it in the set variable.

```
library(ElemStatLearn)
set = training_set
```

In this part of the code the red and green background region is created that can later be observed, this is done through the "by", and 0.01 is interpreted as 0 or 1, and from this is that it is classified as green or as red. The -1 and +1 give the space around the edges, this in order that the points do not stick together so much. The result is shown with respect to the Age and Estimated Salary fields, that is why in the max and min functions the set variable is accessed and the column number is entered.

```
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)
```

In this part, only the “x” and “y” axes of the graph are assigned a name.

```
colnames (grid_set) = c ('Age', 'EstimatedSalary')
```

The background coloring is done here. With the help of the classifier, we try to predict the result of each one of the pixel bits. The type equaled to response is used so that the result is shown as numeric, as opposed to using "class" which shows the label assigned to that value. The ifelse function takes the previous variable and determines that if the value is greater than .5 it is set to 1, otherwise it is defined as 0.

```
prob_set = predict (classifier, type = 'response', newdata = grid_set)
y_grid = ifelse (prob_set> 0.5, 1, 0)
```

The values are presented with the help of the plot function, which allows creating a graph by passing two vectors as reference of the axes with xlim and lim, and the data source being column number 3 of the dataset stored in the variable set. The main title is established with main and that of the “x” and “y” axes with xlab. The result is a point spread without color /

```
plot (set [, -3],
     main = 'Logistic Regression (Training set)',
     xlab = 'Age', ylab = 'Estimated Salary',
     xlim = range (X1), ylim = range (X2))
```

This code creates the limits of the plotted values, in the same way the line between the green and red colors is generated.

```
contour (X1, X2, matrix (as.numeric (y_grid), length (X1), length (X2)),
add = TRUE)
```

Here all the data stored in the variable ypred is reviewed and the ifelse function is used to get coloring the points shown. The pch parameter allows you to choose the type of figure shown in the graph. This would be the code needed to display the analysis of the training set data.

```
points (grid_set, pch = '.', col = ifelse (y_grid == 1, 'springgreen3',
'tomato'))
points (set, pch = 21, bg = ifelse (set [, 3] == 1, ' green4 ', ' red3
'))
```

Viewing the test set results

The steps to construct the graph of the test set data are the same as those shown and explained above, only substituting the source of the data.

```
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')
prob_set = predict (classifier, type = 'response', newdata = grid_set)
y_grid = ifelse (prob_set> 0.5, 1, 0)
plot (set [, -3],
     main = 'Logistic Regression (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
') )
```

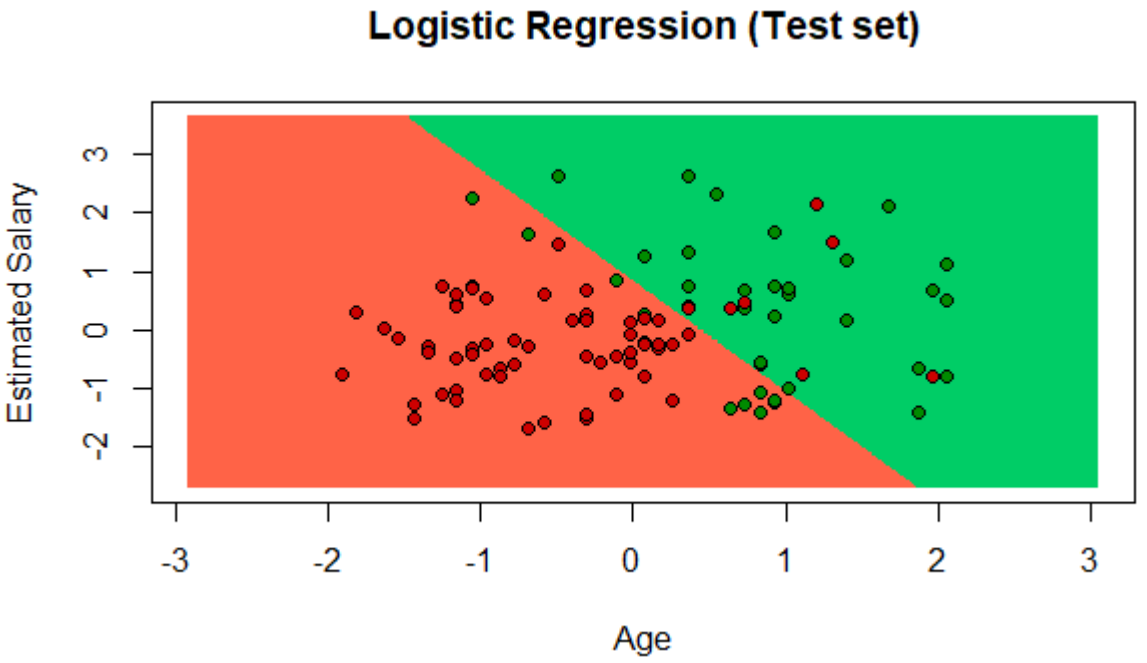
Training set graph conclusion

The center line of the graph is the separation limit of the two classifiers, thus creating the way the current data set is classified. It can be seen that there are two different colors in the scatter of points, and some green are on the red background, and other red on the green background. Points that do not match the background color are predictions made wrong in the training set. The red color indicates that they are customers who did not buy an SUV, while the green region classifies those who did buy the SUV according to the ads on social networks.

Graph 5. Training set

Conclusion of the graph of the test set

It can be observed that the incorrect predictions were not minimized, only that since there is a smaller number of data, fewer results are shown in the graph. When you are analyzing the same set of information, the red region continues to represent customers who did not buy the SUV, while the green region indicates customers who did.



Graph 6. Test set