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

Práctica 3

Landa Alvarez Ariel Nicolas 17211531
Ceron Uribe Arturo #17211506

MC. JOSE CHRISTIAN ROMERO HERNANDEZ

Campus Tomas Aquino

Antes de comenzar con el desarrollo del programa es necesario establecer el directorio de trabajo y la lectura de los datos del archivo, esto se hace de la siguiente manera.

```
getwd()
setwd("/home/chris/Documents/itt/Enero_Junio_2020/Mineria_de_datos
/DataMining/MachineLearning/LogisticRegression")
getwd()

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

Debemos también separar el dataset en 2 partes, uno que funcionara de entrenamiento para el modelo, y otro que funcionara para obtener la precisión del modelo.

```
# 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)
```

Preparamos los datos realizando un scale, que nos ayudara a normalizar los datos que se encuentran en el data set para generar un index y con esto, preparamos un modelo clasificador lineal (General Lineal Regression) especificando la formula, que es el atributo a buscar.

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

# Fitting Logistic Regression to Training set
classifier = glm(formula = Purchased ~ .,
                 family = binomial,
                 data = training_set)
```

Teniendo el modelo, realizamos el entrenamiento utilizando el dataset test, para obtener un acercamiento de nuestra variable, en este caso al tratarse de regresión logística, los valores deben ser 1 y 0 por lo que separamos las predicciones en estos valores utilizando como margen 0.5. Podemos generar una matriz de confusión(Confusion Matrix) la cual tiene los valores reales y compara a la predicción realizada por el modelo, dando a entender que de 64 datos 0, el programa calculo 67 como 0.

```
  y_pred
    0  1
0 57  7
1 10 26
```

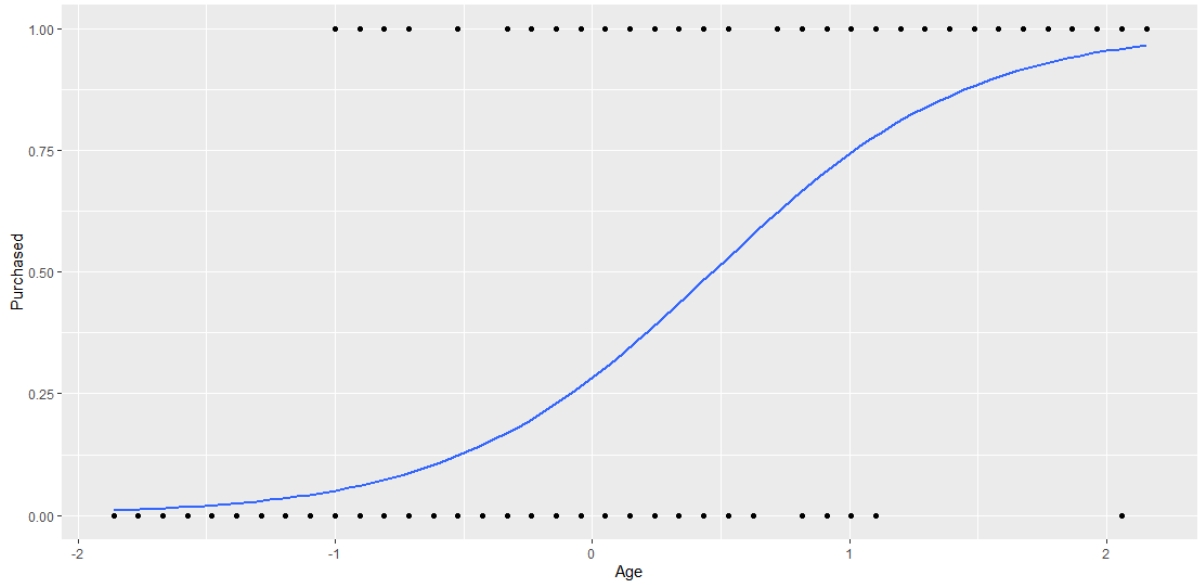
```
# Predicting the Test set results
prob_pred = predict(classifier, type = 'response', newdata =
test_set[-3])
prob_pred
y_pred = ifelse(prob_pred > 0.5, 1, 0)
y_pred

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

Ahora imprimimos la gráfica de los datos, para esto vamos a comparar cada una de las variables no dependientes a la variable dependiente, de otra forma dicha, compararemos cada x con nuestra y.

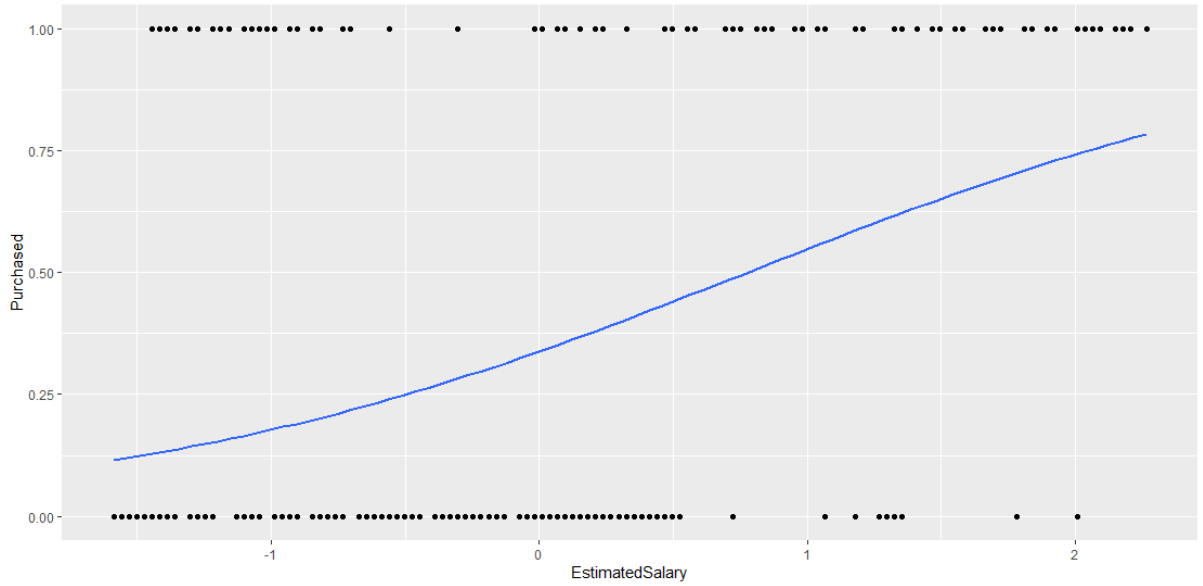
```
#
library(ggplot2)
ggplot(training_set, aes(x=EstimatedSalary, y=Purchased)) +
geom_point() +
  stat_smooth(method="glm", method.args=list(family="binomial"),
se=FALSE)

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

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

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

```
se=FALSE)
```

se compara la edad y si se compro o no, de esta grafica se puede rescatar que existe una relación entre la edad de la persona y si compra o no.

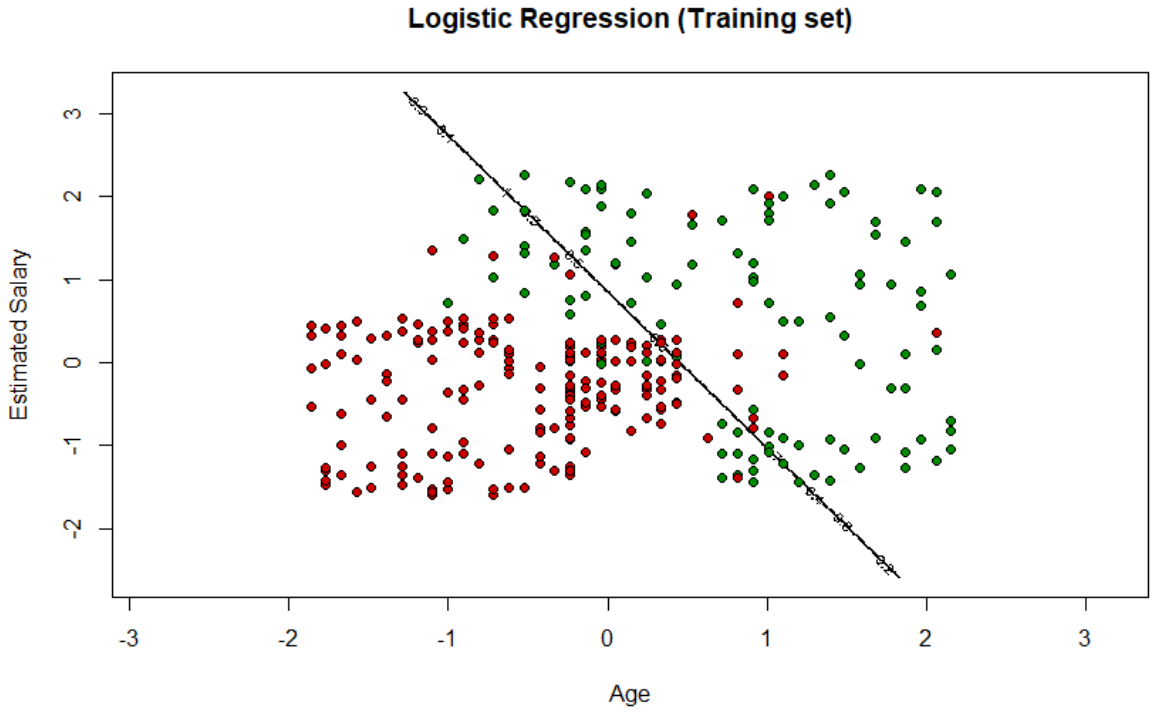


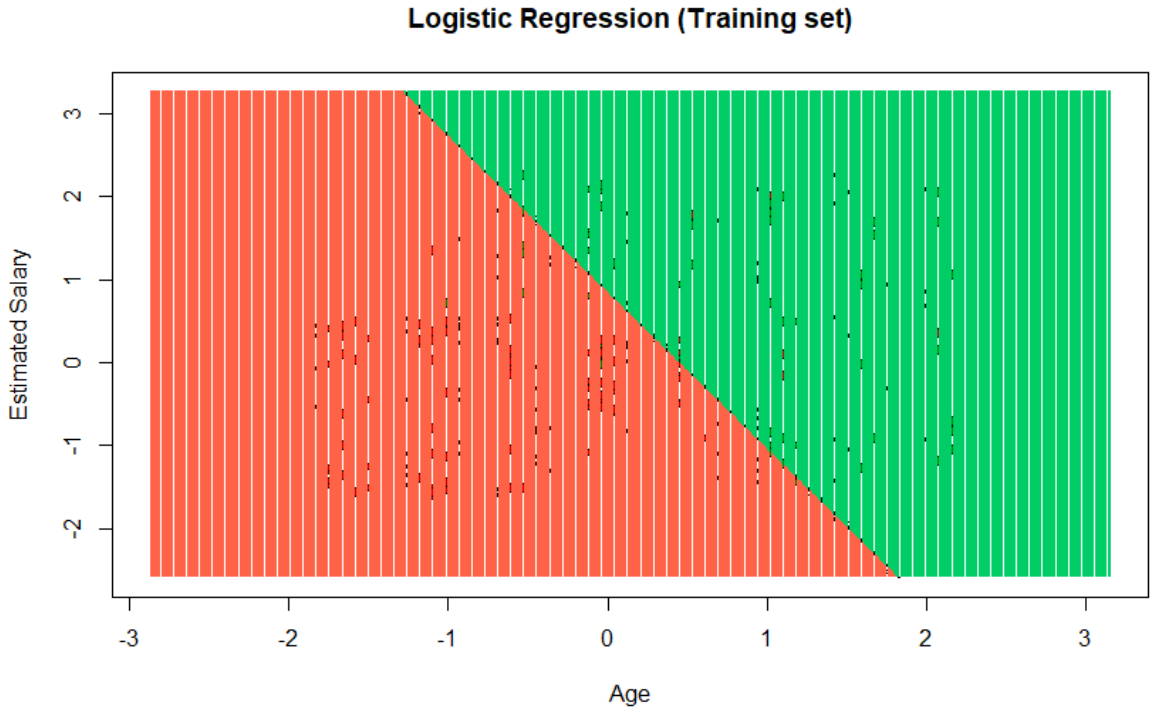
Se compara el salario y si se compro o no.
Se puede observar en esta gráfica que un salario alto tiene más probabilidad de comprar pero esta declaración puede ser ambigua ya que existen muchos sujetos de un sueldo bajo que compran.



```
# Visualization the Training set result
# install.packages('ElemStatLearn') No work for me,
# manual mode. Go to this URL
https://cran.r-project.org/src/contrib/Archive/ElemStatLearn/
# Download with the latest date 2019-08-12 09:20 12M
# Then follow this page steps
https://riptutorial.com/r/example/5556/install-package-from-local-
source

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')
prob_set = predict(classifier, type = 'response', newdata =
grid_set)
y_grid = ifelse(prob_set > 0.5, 1, 0)
plot(set[, -3],
     main = 'Logistic Regression (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 = '.', bg = ifelse(y_grid == 1,
'springgreen3', 'tomato'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4',
'red3'))
```

Se puede graficar el modelo utilizando nuestro modelo clasificador, realizamos el mismo procedimiento realizado anteriormente realizando la comparación entre los datos y los datos arreglados para el modelo. En esta grafica podemos observar la comparación entre los datos reales, la línea que los separa y posteriormente con color para ver la predicción que realiza el modelo.





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

Este caso es similar, con la diferencia de que los datos que utilizamos es test_set y lo comparamos a el modelo con los datos de train_set

El Resultado es la siguiente grafica, que contiene los puntos de los datos de prueba y la linea generada por el modelo entrenado.