EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLOGICO NACIONAL DE MEXICO.

National Technology of Mexico
Technological Institute of Tijuana

ACADEMIC SUBDIRECTION
Systems and Computing Department

SEMESTER
February – June 2021

CAREER
Information and Communication Technologies Engineer

SUBJECT AND KEY:
Data Mining BDD-1703TI9A

STUDENT'S NAME AND REGISTRATION:

Camacho Manabe Juan Daniel	17210534
Velázquez Farrera César Alejandro	17212937

NAME OF THE JOB:
Practice 5

UNIT TO BE EVALUATED
Unit III

TEACHER’S NAME:
Mc. José Christian Romero Hernández

Practice #4 SVM

Instructions

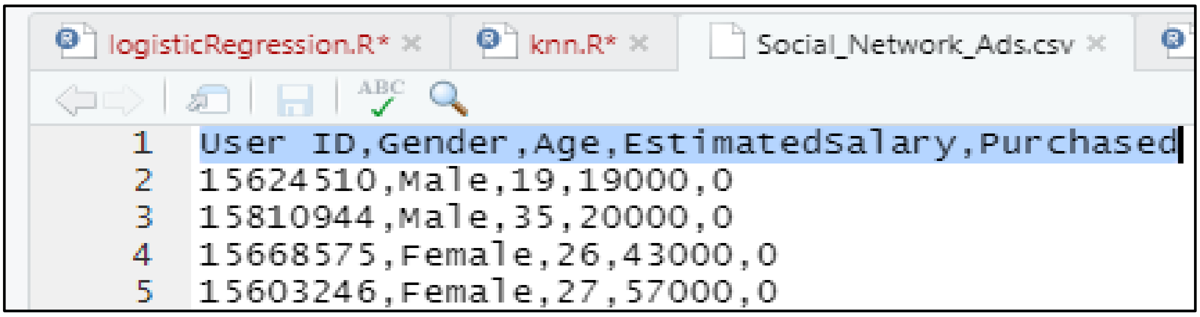
Make the analysis corresponding to the support vector machine R script, which must be documented in its repository by putting in it its visual results and its detailed description of its observations as well as the source of the code.

We can define the path we want to work on.

```
getwd()
setwd("/home/chris/Documents/itt/Enero_Junio_2020/Mineria_de_datos/DataMinin
g/MachineLearning/SVM")
getwd()
```

We look for our data file on the computer with the help of the file explorer, if we are working in the folder where the file is located, we simply load it with the name of the file.

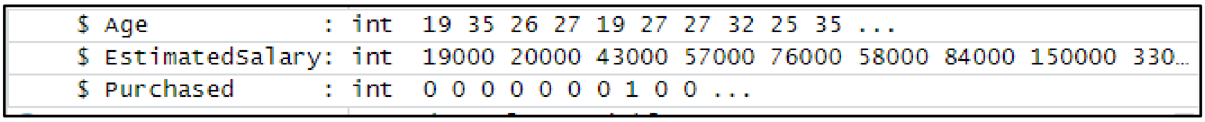
```
dataset = read.csv(file.choose())
dataset = read.csv('Social_Network_Ads.csv')
```



```
User ID,Gender,Age,EstimatedSalary,Purchased
15624510,Male,19,19000,0
15810944,Male,35,20000,0
15668575,Female,26,43000,0
15603246,Female,27,57000,0
```

Since the data file has columns that will not be useful, we only select the columns that we want to work with.

```
dataset = dataset[3:5]
```



```
 $ Age            : int  19 35 26 27 19 27 27 32 25 35 ...
 $ EstimatedSalary: int  19000 20000 43000 57000 76000 58000 84000 150000 330…
 $ Purchased      : int  0 0 0 0 0 0 0 1 0 0 ...
```

We load the library "caTools" that has several functions for statistics.
With the function "set.seed (n)" we sow a randomness seed where "n" is the starting point used in the generation of the sequence of random numbers.

```
library(caTools)
set.seed(123)
```

Next, we divide the data in the “Purchased” column, with a ratio of 0.75, that is, 75% of the data will be taken as true and the remaining 25% as false.
Later we take the values of the division of the data that were true and we use them for training, the rest of the data was labeled as false will be used to test the effectiveness of the trained model with the data labeled as true.

```
split = sample.split(dataset$Purchased, SplitRatio = 0.75)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)
```

For training and testing it is necessary to normalize the data, so we use the "scale" function. The characteristic scale is a method used to normalize the range of independent variables or characteristics of the data. In data processing, it is also known as data normalization and is usually done during the data pre-processing step.

```
training_set[-3] = scale(training_set[-3])
test_set[-3] = scale(test_set[-3])
```

We install and load the library (“e1071”), which contains statistics functions, it is currently depreciated for the most recent versions of r.

```
Install.packages('e1071')
library(e1071)
```

We adopt the vector support machine to the training data set, that is, the classification with the "svm" method which is used to fit the model. A curious fact is that it uses a kernel.

```
classifier = svm(formula = Purchased ~ .,
                 data = training_set,
                 type = 'C-classification',
                 kernel = 'linear')
```

Obtaining the probability of predicting correct classifications with the test data. We print the prediction. We evaluate if the probability of the prediction is between 0.5, 1 and 0 and print the probability.

```
y_pred = predict(classifier, newdata = test_set[-3])
y_pred
```

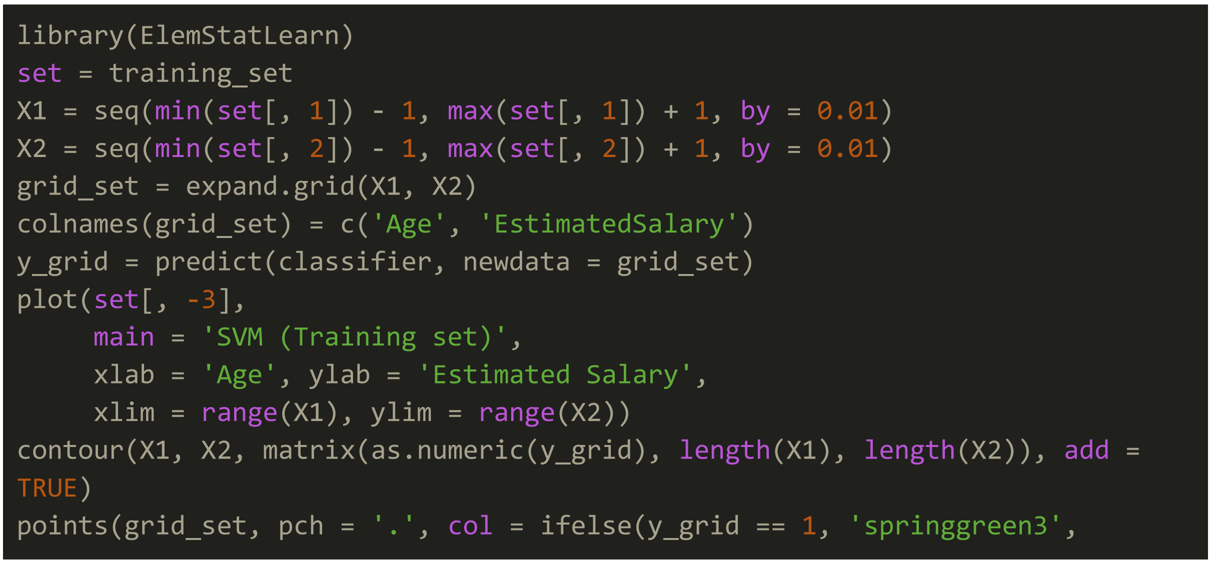
We save the resulting data in a table with the test data and the corresponding predictions, that is, we create a confusion matrix. Then we print it.

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

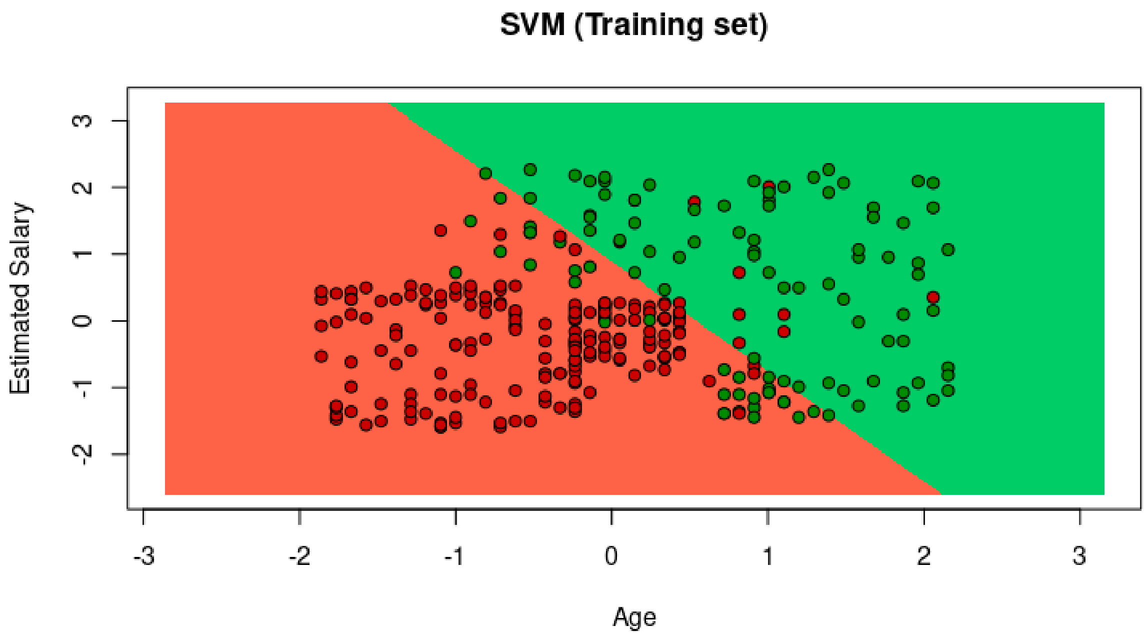
To continue we will need a special package that contains the "ElemStatLearn" library, since the package is obsolete for new versions of r and does not come by default in the rStudio tools, it is necessary to download and install it manually. The file will be found in the following link: https://cran.r-project.org/src/contrib/Archive/ElemStatLearn/, the installation will be done with the following command, substituting the appropriate file address.

```
install.packages("~/home/daniel-
camacho/Descargas/ElemStatLearn_2015.6.26.tar.gz", repos=NULL,
type="source")
```

1. Once the package is installed, we load the "ElemStatLearn" library.
2. With the command "set" we take the training data, then we create two variables "x1", "x2" where we use the function "seq" (sequence), we will define a minimum and maximum range, we will add a positive integer and multiply it by the length of "0.01"
3. Next we define a grid that we expand using the dimensions defined in "x1" and "x2".
4. We define the columns of the grid using a vector, which will be the year and estimated salary.
5. Next we test the logistic regression model but with the grid that contains the columns that we define.
6. We graph the data with a maximum of -1, we put a title to the graph that is inside (grid), we name the axes, we assign the limits for the axes keeping the data within the dimensions of the grid.
7. We place an outline and add a range of numbers conditional on the minimum and maximum numerical range of the grid.
8. We add points to the graph comparing the data of the year and estimated salaries with the predictions of the logistic regression model, where if they are different, a red color will be assigned and if they are true a green one, in the case of errors, red points will be placed within the green area that has been defined as true sets and, on the contrary, a green point in the area defined as false.

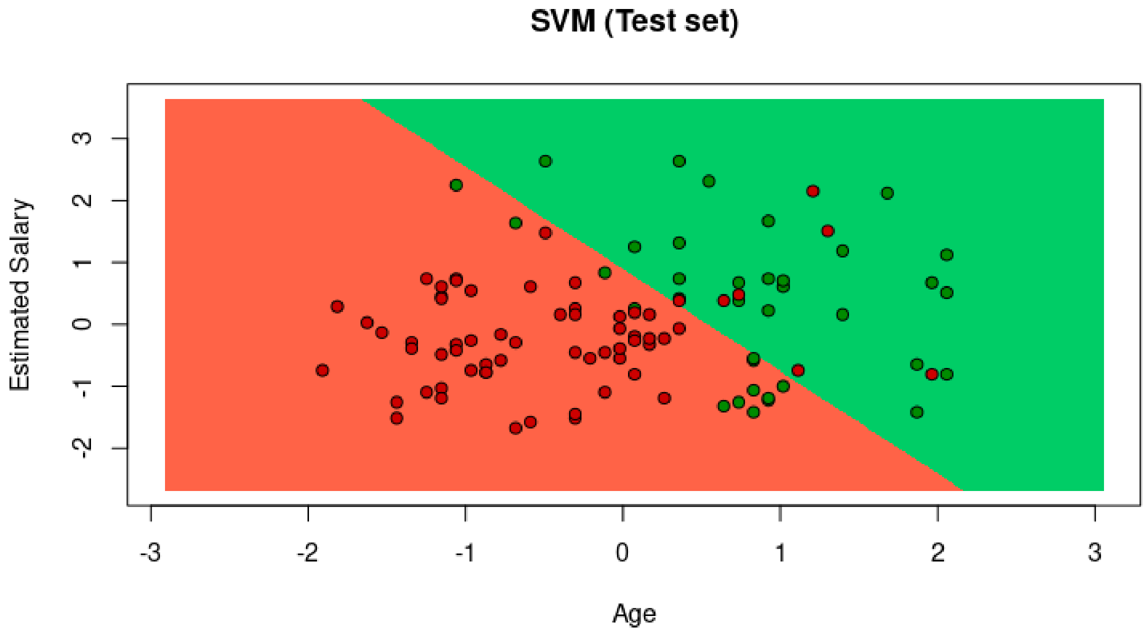


```
library(ElemStatLearn)
set = training_set
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'EstimatedSalary')
y_grid = predict(classifier, newdata = grid_set)
plot(set[, -3],
     main = 'SVM (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'))
```



In the same way as in the previous code, the evaluation and classification of the data is carried out, only this time, for the graph it will not be the training data but the test data, for which the data is assigned of the variable "test_set" obtained in the division of the data corresponding to 25% of the totality of the dataset that contains the three initial columns.

```
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, newdata = grid_set)
plot(set[, -3], main = 'SVM (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'))
```



Conclusiones y observaciones

Based on the observations, the first method is the easiest to implement in terms of syntax and understanding of the R language. However, when representing the results of the predictions (using the confusion matrix and), it is not easy to understand at first glance for users who are unfamiliar with the subject.

With the results obtained with the predictions, it can be concluded that the columns of age (“age”) have a higher correlation than the columns of estimated salary (“EstimatedSalary”) and Purchased (“Purchased”). This is evident in the graphs of points with the method "svm" by the line, which represents the vector support, although in two dimensions it is not appreciated in the way that it would really be seen, for example in third dimension.

The second method, with the use of the "ElemStatLearn" library, can generate a graph much easier to understand, at the cost of complex syntax. It can be seen in the second graph of the second method, the red points represent the people who did not buy the product and the green ones those who did.

We can conclude that the older the potential customer is, the more likely they are to sell the product. Unlike the customer being younger, there is less chance that he will purchase the product.

Bibliography

1. jcromerohdz. (2020). LogisticRegression. 29/05/30, de GitHub Sitio web: https://github.com/jcromerohdz/DataMining/tree/master/MachineLearning/SVM

2. https://cran-archive.r-project.org/web/checks/2020/2020-01-28_check_results_ElemStatLearn.html

3. https://cran.r-project.org/src/contrib/Archive/ElemStatLearn/

4. https://riptutorial.com/r/example/5556/install-package-from-local-source

5. Escamilla, D. V., Escamilla, C., & Rodríguez, C. A. (2015). Eye Blink Detection Using a Support Vector Machine Classifier. In *Memorias del Congreso Nacional de Ingeniería Biomédica* (Vol. 2, No. 1, pp. 108-111).

6. Flores, A., Maldonado, S., & Weber, R. (2015). Selección de atributos y Support Vector Machine adaptado al problema de fuga de clientes. *Revista Ingeniería de Sistemas*, *29*.

7. Barbona, I., & Beltrán, C. (2016). Método de clasificación supervisada support vector machine: una aplicación a la clasificación automática de textos.

8. Oliver-Muncharaz, J. (2018). Análisis y clasificación de indicadores técnicos mediante support vector machine. *Finance, Markets and Valuation*, *4*(1), 81-93.

9. Campos-Zárate, P. A., Hurtado, E. G., & Ortega, J. C. P. (2020). Reconocimiento facial basado en Least-Squares Support Vector Machine y características LNDP. *Res. Comput. Sci.*, *149*(8), 841-851.