**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 2*

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.

| setwd("C:/Users/CORSAIR/DataminingRomero/DataMining/MachineLearning/MultipleLinearRegression")  getwd()  # Importing the dataset dataset <- read.csv('50\_Startups.csv') |
| --- |

Posteriormente se hace una codificación de los estados dentro del dataset, se cambian los nombres por valores numéricos.

| # Encoding categorical data  dataset$State = factor(dataset$State,  levels = c('New York', 'California', 'Florida'),  labels = c(1,2,3)) |
| --- |

Se comienza la preparación de los valores de prueba y entrenamiento necesarios para la utilización del modelo.

| # Splitting the dataset into the Training set and Test set split <- sample.split(dataset$Profit, SplitRatio = 0.8) training\_set <- subset(dataset, split == TRUE) test\_set <- subset(dataset, split == FALSE) |
| --- |

Se crea una variable regressor donde se utilizara la función lm para modelos lineales, dando como parámetros el profit como su fórmula a utilizar y los datos de entrenamiento que se prepararon previamente

| # Fitting Multiple Linear Regression to the Training set #regressor = lm(**formula** = Profit ~ R.D.Spend + Administration + Marketing.Spend + State) regressor = lm(**formula** = Profit ~ .,  data = training\_set ) summary(**regressor**) |
| --- |

Se realiza una predicción de los resultados utilizando el regressor creado y los datos de prueba que se generaron anteriormente.

| # Prediction the Test set results y\_pred = predict(regressor, newdata = test\_set) |
| --- |

Se reescribe el regressor utilizando eliminación hacia atrás, esto consiste en ir ejecutando la función y cada vez ir eliminando una de las variables que toma en cuenta, esto siempre con los datos completos del dataset.

| # Assigment: visualize the siple liner regression model with R.D.Spend   # Building the optimal model using Backward Elimination regressor = lm(formula = Profit ~ R.D.Spend + Administration + Marketing.Spend + State,  data = dataset ) summary(regressor)  regressor = lm(formula = Profit ~ R.D.Spend + Administration + Marketing.Spend,  data = dataset ) summary(regressor)  regressor = lm(formula = Profit ~ R.D.Spend + Marketing.Spend,  data = dataset ) summary(regressor)  regressor = lm(formula = Profit ~ R.D.Spend + Marketing.Spend,  data = dataset ) summary(regressor) |
| --- |

Una vez construido el modelo óptimo en la variable regressor se realiza una nueva prediccion de resultados utilizando los datos de prueba preparados.

| y\_pred = predict(regressor, newdata = test\_set) |
| --- |

Generación de la funcion backwardElimination, función utilizada que recibe dos valores como parámetros. X como los valores de entrenamiento y SL como el valor que no debe de superar para tener un modelo óptimo.   
Esta función obtiene los coeficientes del modelo de la variable regressor que se tiene, obteniendo valores máximos, lo que se busca es que estos valores máximos no sean mayores al SL ya que los resultados buscados es obtener un valor cercano al 0 o que sea 0 para que funcione la regresión. Una vez que se termine de ejecutar el ciclo encargado de la verificación del valor máximo se debe de obtener un regressor con el modelo más óptimo para la ejecución del programa.

| # Homework analise the follow atomation backwardElimination function  backwardElimination <- function(x, sl) {  numVars = length(x)  for (i in c(1:numVars)){  regressor = lm(formula = Profit ~ ., data = x)  maxVar = max(coef(summary(regressor))[c(2:numVars), "Pr(>|t|)"])  if (maxVar > sl){  j = which(coef(summary(regressor))[c(2:numVars), "Pr(>|t|)"] == maxVar)  x = x[, -j]  }  numVars = numVars - 1  }  return(summary(regressor)) } |
| --- |

#dataset = dataset[, c(1,2,3,4,5)]

Finalmente se hace uso de la función enviando los datos de prueba como parámetros y el SL deseado para que tenga un mínimo que obtener.

| training\_set backwardElimination(training\_set, SL) |
| --- |

Los resultados obtenidos son los siguientes, demostrando así que los coeficientes obtenidos son 0 o muy cercanos a 0, esto se puede identificar por la cantidad de ‘\*’ que tiene cada variable dentro de la sección “Coefficients”.

