CENTRO DE CIENCIAS BÁSICAS

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

APRENDIZAJE INTELIGENTE

6° "A"

ACTIVIDAD 3.1: REGRESIÓN LINEAL EN PYTHON

Profesor: Francisco Javier Luna Rosas

Alumnos:

Espinoza Sánchez Joel Alejandro

Gómez Garza Dariana

González Arenas Fernando Francisco

Fecha de Entrega: Aguascalientes, Ags.**, 12** de mayo de 2021

Actividad 3.1: Regresión Lineal en Python

Objetivo:

Mediante el desarrollo de esta práctica, implementar el análisis de regresión lineal en Python.

Introducción:

En estadística, la regresión lineal o ajuste lineal es un modelo matemático usado para aproximar la relación de dependencia entre una variable dependiente Y y m variables independientes X_i con $m \in \mathbb{Z}^+$y un término independiente ε.

Este modelo es representado como:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_m X_m + \varepsilon$$

La regresión lineal enfatiza la relación estadística entre dos variables continuas conocidas como variables de predicción y respuesta. Cuando hay más de una variable predictora, se convierte en regresión lineal múltiple.

La variable predictora se denota con mayor frecuencia como x y también se conoce como variable independiente.

La variable de respuesta se denota con mayor frecuencia como y y también se conoce como variable dependiente.

La relación entre dos variables se llama determinista si una variable determina perfectamente la otra variable, de lo contrario se le nombrará como no determinista.

Pregunta de Investigación:

¿Cómo se puede implementar regresión lineal para múltiples variables dependientes en Python?

Predicción:

Creemos que la implementación consistirá en buscar adecuar para Python la regresión lineal mediante los cálculos básicos que este modelo requiere sin mucho esfuerzo.

Materiales:

Una computadora con Python y el entorno Anaconda.

Método (Variables):

Dependiente: La recta de regresión que se obtendrá.
Independiente: Los datos a tratar con el conjunto de datos de cáncer.
Controlada: El algoritmo de regresión lineal a implementar.

Seguridad:

Realmente no se trabajó en campo, por lo que no se corren riesgos con la práctica.

Procedimiento:

1.- Se importó la librería de trabajo para extracción de datos de Python:

```
# %% Código
# Uso de librerías
import pandas as pd
```

2.- Se introdujeron los datos del archivo csv:

```
# Extracción de archivos
ArchivoCSV = pd.read_csv('C:/Users/alexe/Desktop/datos/prostate_data.csv',
                         engine = 'python', sep=';', decimal=',')
DatosMuestra = ArchivoCSV.values.tolist()
```

3.- Se inicializaron variables que guardarían los datos necesarios para los cálculos de la regresión lineal:

```
# Inicialización de variables
Componentes = []
n = len(DatosMuestra)
Sx = 0
Sx2 = 0
Sy = 0
Sxy = 0
```

4.- Se realizó cíclicamente el análisis de cada variable con relación a la que se deseaba predecir:

```
# Obtención de las sumatorias de los datos
for i in range(8):
    for j in range(n):
        Sx = Sx + DatosMuestra[j][i]
        Sx2 = Sx2 + (DatosMuestra[j][i] * DatosMuestra[j][i])
        Sy = Sy + DatosMuestra[j][8]
        Sxy = Sxy + (DatosMuestra[j][i] * DatosMuestra[j][8])

    a = ((n * Sxy) - (Sx * Sy))/((n * Sx2) - (Sx * Sx))
    b = ((Sy) - (a * Sx))/(n)
    Componentes.append([a,b])
print(Componentes)
```

5.- Se imprimieron los resultados:

```
# Impresión de los resultados
print("RSS = (y1 - " + str(Componentes[0][0]) + " - " + str(Componentes[0][1]) + "x1)^2")
for i in range(7):
    print("+ (y" + str(i + 2) + " - " +
          str(Componentes[i + 1][0]) + " - " + str(Componentes[i + 1][1]) + "x" +
          str(i + 2) + ")^2")
```

Obtención y Procesamiento de Datos:

Al ejecutar el código, el programa imprime un arreglo donde guarda el coeficiente tanto independiente como el de la pendiente de cada variable según el orden del archivo dado:

```
[[0.7193203917677407, 1.5072974580261764],
[1.3515046532360984, -1.7723033642754538],
[0.574535558892638, -32.11870063115975],
[0.8479231914508336, -48.54676046837955],
[1.0818944628431089, -62.43381467367806],
[1.4084460918147805, -82.28769364755533],
[0.7307614863334922, -37.99550849964557],
[0.387057920391631, -18.923764919755822]]
```

Sin embargo, para una mejor presentación de los datos, estos mismos salen en forma de ecuación como la recta de regresión lineal posteriormente:

```
RSS = (y1 - 0.7193203917677407 - 1.5072974580261764x1)^2
+ (y2 - 1.3515046532360984 - -1.7723033642754538x2)^2
+ (y3 - 0.574535558892638 - -32.11870063115975x3)^2
+ (y4 - 0.8479231914508336 - -48.54676046837955x4)^2
+ (y5 - 1.0818944628431089 - -62.43381467367806x5)^2
+ (y6 - 1.4084460918147805 - -82.28769364755533x6)^2
+ (y7 - 0.7307614863334922 - -37.99550849964557x7)^2
+ (y8 - 0.387057920391631 - -18.923764919755822x8)^2
```

Conclusiones:

Joel Alejandro Espinoza Sánchez: La regresión lineal nos ayuda en el análisis de correlación de variables que es muy útil. En este trabajo, nos habían encargado adaptar este modelo a Python que es muy útil para observar cómo se puede tratar este modelo que se ha revisado en temas de estadística pero es sencillo de tratar cuando se elabora por medio de un algoritmo computacional.

Dariana Gómez Garza: A mi parecer esta práctica fue un poco diferente a las anteriores, ya que tuvimos que basarnos más a una fórmula para realizar el procedimiento y obtener los resultados. Así que esta práctica fue tal vez un poco

más "sencilla" de realizar, además que con la base teórica, el ejercicio visto en clase y un poco de guía en programa en R pudimos darnos una mejor idea de cómo realizar el trabajo en Python.

Fernando Francisco González Arenas: La regresión lineal es un método muy útil en el aprendizaje supervisado para la predicción de variables. Con este método se puede predecir una variable dependiente Y con una variable independiente X con ayuda de 2 parámetros, los cuales son básicamente la pendiente y la ordenada al origen de la recta, los cuales se puede obtener el óptimo utilizando la formula del método de ajustar por mínimos cuadrados. este método es muy útil para predecir, incluso existen variaciones o extensiones de la regresión lineal en el aprendizaje supervisado para la predicción de variables en problemas específicos. es muy importante comprender muy bien este método, ya que es muy utilizado en este tipo de aprendizaje con modificaciones, pero es prácticamente el mismo principio de regresión lineal.

Referencias:

Equipo de MathWorks. (2016). *¿Qué es la regresión lineal?* Mayo 10, 2021, de MathWorks Sitio web: https://es.mathworks.com/discovery/linear-regression.html

Noren, A. (2019). *Correlación y regresión lineal simple.* Mayo 8, 2021, de Sitio Big Data Sitio web: https://sitiobigdata.com/2019/10/25/que-es-la-regresion-lineal/

Anexos:

Anexo 1: Código del programa en lenguaje Python:

```
#%% Presentación
'''
   Universidad Autónoma de Aguascalientes

         Centro de Ciencias Básicas
 Departamento de Ciencias de la Computación
          Aprendizaje Inteligente
                   6° "A"

                Actividad 3.01

   Profesor: Francisco Javier Luna Rosas
  Alumnos:
      Espinoza Sánchez Joel Alejandro
      Gómez Garza Dariana
      González Arenas Fernando Francisco

  Fecha de Entrega: 12 de mayo del 2021

Descripción: Regresión lineal en Python
'''


# %% Código
# Uso de librerías
import pandas as pd

# Extracción de archivos
ArchivoCSV = pd.read_csv('C:/Users/alexe/Desktop/datos/prostate_data.csv',
                         engine = 'python', sep=';', decimal=',')
DatosMuestra = ArchivoCSV.values.tolist()

# Inicialización de variables
Componentes = []
n = len(DatosMuestra)
Sx = 0
Sx2 = 0
Sy = 0
Sxy = 0

# Obtención de las sumatorias de los datos
for i in range(8):
    for j in range(n):
        Sx = Sx + DatosMuestra[j][i]
        Sx2 = Sx2 + (DatosMuestra[j][i] * DatosMuestra[j][i])
        Sy = Sy + DatosMuestra[j][8]
        Sxy = Sxy + (DatosMuestra[j][i] * DatosMuestra[j][8])
```

```
    a = ((n * Sxy) - (Sx * Sy))/((n * Sx2) - (Sx * Sx))
    b = ((Sy) - (a * Sx))/(n)
    Componentes.append([a,b])
print(Componentes)

# Impresión de los resultados
print("RSS = (y1 - " + str(Componentes[0][0]) + " - " + str(Componentes[0][1]) +
"x1)^2")
for i in range(7):
    print("+ (y" + str(i + 2) + " - " +
          str(Componentes[i + 1][0]) + " - " + str(Componentes[i + 1][1]) + "x"
+
          str(i + 2) + ")^2")
```