UNIVERSIDAD MAYOR REAL Y PONTIFICIA DE SAN FRANCISCO XAVIER DE CHUQUISACA

Informe Laboratorio 7

Regresión Lineal Simple

UNIVERSITARIA: Ortube Rengel Erika Mariana

CARRERA: Ingeniería de Sistemas

MATERIA: Inteligencia Artificial I (SIS420)

DOCENTE: Ing. Carlos W. Pacheco Lora

SUCRE – BOLIVIA

Regresión Lineal Simple

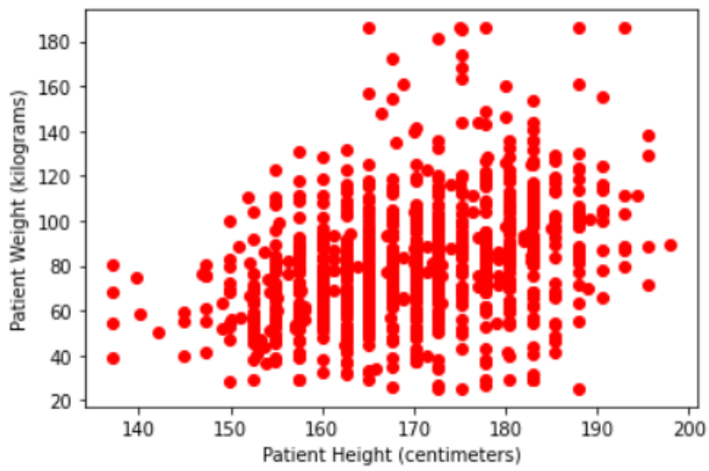
Se recopilaron datos de un *dataset* sobre sobre la altura y peso corporal de 1000 pacientes ingresados en un hospital. Para tratar de predecir si mientras más altos (as), más bajos o una altura promedio, llegan a tener un peso bajo, peso promedio o peso alto/muy alto.

Lectura de datos

```
[25] #Datos sobre las alturas y peso corporal de 1000 pacientes ingresados en un hospital
     #Leer datos
     data = np.loadtxt('dataset.txt', delimiter=',')
     X, y = data[:, 0], data[:,1]
     m = y.size
```

Definición de función para graficar los datos

```
[27] def graficarDatos(x, y):
       fig = pyplot.figure()
       pyplot.plot(x, y, 'ro')
       pyplot.xlabel('Patient Height (centimeters)')
       pyplot.ylabel('Patient Weight (kilograms)')
```



Como podemos observar la altura de la mayoría de los pacientes oscila entre 155 – 185 centímetros y su peso oscila entre 50 – 90 kilogramos.

Agregando una columna de 1s en la misma cantidad de m (número de ejemplos) y los coloca delante de las x. Matriz de mx2.

```
[29] #No olvidarse agregar '1s' a la matriz de x
     X = np.stack([np.ones(m), X], axis=1)
```

```
[30] X
```

```
array([[  1. , 180.3],
       [  1. , 160. ],
       [  1. , 172.7],
       ...,
       [  1. , 160. ],
       [  1. , 162.6],
       [  1. , 165.1]])
```

Cálculo del costo

```
[57] def calcularCosto(X, y, theta):
       m = y.size
       J = 0
       h = np.dot(X, theta)
       # print(h)
       J = (1/(2 * m)) * np.sum(np.square(np.dot(X, theta) - y))
       return J
```

```
theta=np.array([0.1, 0.5])
JJ = calcularCosto(X, y, theta)
print(f"con theta:{ theta } se obtiene un costo de: {JJ}")
```

```
con theta:[0.1 0.5] se obtiene un costo de: 317.1943689499999
```

Se observa que dando valores de [0.1, 0.5] para theta, se llega al costo de *317.19*, pero con valores de [0.1, 0.3] se llega a un de *764.14*, si bien el costo mínimo no es cercano a cero, es una alternativa para el costo.

Calculo Descenso por el Gradiente para encontrar los valores de theta que permitan minimizar aún más el costo

```
[88] def calcularDescensoGradiente(X, y, theta, alpha, numero_iteraciones):
       m = y.shape[0]
       theta = theta.copy()
       J_historico = []

       for i in range(numero_iteraciones):
         theta = theta - (alpha / m) * (np.dot(X, theta) - y).dot(X)
         J_historico.append(calcularCosto(X, y, theta))

       return theta, J_historico
```

```
theta = np.zeros(2)

num_ite = 200000
alpha = 0.00006

theta, J_historico = calcularDescensoGradiente(X, y, theta, alpha, num_ite)
print(f"los valores de theta calculados son: { theta }")
print(f"con un costo de: { J_historico[-1]} ")
```

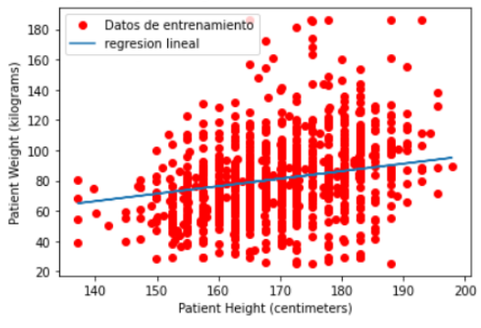
```
los valores de theta calculados son: [-3.10027935  0.49612997]
con un costo de: 308.94823406788186
```

Si bien el costo llega a bajar, aun no es cercano a 0.

Grafico de la Regresión Lineal

```
graficarDatos(X[:, 1], y)
pyplot.plot(X[:,1], np.dot(X, theta), '-')
pyplot.legend(['Datos de entrenamiento', 'regresion lineal'])
```

```
<matplotlib.legend.Legend at 0x7f01c45a85d0>
```



Predicción para y (peso) con una altura de 194.3 (X)

```
[120] y_pred = np.dot([1, 194.3], theta)
      print(y_pred)
```

```
93.29777395395402
```