UNIVERSIDAD MAYOR, REAL Y PONTIFICIA DE SAN FRANCISCO XAVIER DE CHUQUISACA

FACULTAD DE TECNOLOGÍA

INGENIERIA DE SISTEMAS

LABORATORIO 9

ESTUDIANTE: Calle Perez Edwin

CARRERA: INGENIERIA DE SISTEMAS

MATERIA: INTELIGENCIA ARTIFICIAL (SIS 420)

SUCRE – BOLIVIA

Laboratorio 9

Desarrollar lo siguiente:
Al dataset asignado aleatoriamente, aplicar el cuadernillo empleado en clase para regresiones lineales múltiples y aplicar el mismo para identificar, los mejores valores de thetha, el menor valor de la funcion de costo, el número de iteraciones suficientes para alcanzar los valores ideales de theta, el mejor valor de alfa (coheficiente de aprendizaje) y realizar la grafica del comportamiento del costo en funcion al numero de iteraciones y verificar los resultado con la ecuacion de la normal.

Se debe incluir el codigo y la direccion del repositorio empleado de manera obligatoria.
un dataset con al menos m > 200 y n > 5 relacionado a algun fenomeno o comportamiento que se pueda representar por una regresion logistica, a partir del codigo fuente utilizado en clase genera el respetivo modelo.

Dirección de Dataset:

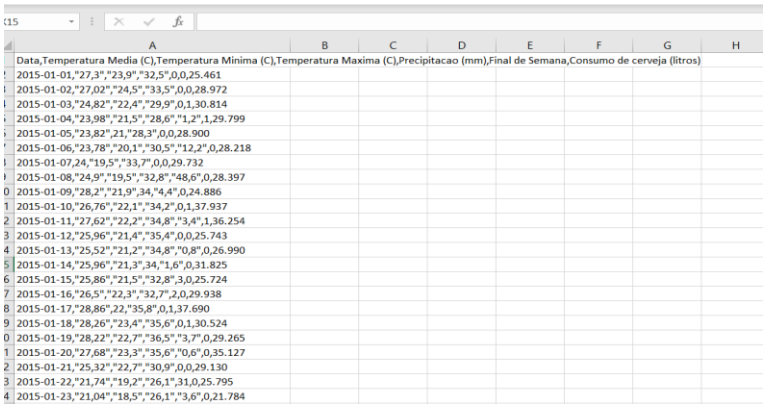
https://www.kaggle.com/datasets/dongeorge/beer-consumption-sao-paulo

Consumo de Cerveza - São Paulo

Los datos (muestra) fueron recolectados en São Paulo — Brasil, en un área universitaria, donde hay algunas fiestas con grupos de estudiantes de 18 a 28 años (promedio). El conjunto de datos utilizado para esta actividad tiene 7 atributos, siendo un objetivo, con un período de un año.

2015-01-01	27,3	23,9	32,5	0	0	25.461
2015-01-02	27,02	24,5	33,5	0	0	28.972
2015-01-03	24,82	22,4	29,9	0	1	30.814
2015-01-04	23,98	21,5	28,6	1,2	1	29.799
2015-01-05	23,82	21	28,3	0	0	28.900
2015-01-06	23,78	20,1	30,5	12,2	0	28.218
2015-01-07	24	19,5	33,7	0	0	29.732
2015-01-08	24,9	19,5	32,8	48,6	0	28.397
2015-01-09	28,2	21,9	34	4,4	0	24.886

Dataset descargado archivo .csv



	A	B	C	D	E	F	G	H
	Data,Temperatura Media (C),Temperatura Minima (C),Temperatura Maxima (C),Precipitacao (mm),Final de Semana,Consumo de cerveja (litros)							
	2015-01-01,"27,3","23,9","32,5",0,0,25.461							
	2015-01-02,"27,02","24,5","33,5",0,0,28.972							
	2015-01-03,"24,82","22,4","29,9",0,1,30.814							
	2015-01-04,"23,98","21,5","28,6","1,2",1,29.799							
	2015-01-05,"23,82",21,"28,3",0,0,28.900							
	2015-01-06,"23,78","20,1","30,5","12,2",0,28.218							
	2015-01-07,24,"19,5","33,7",0,0,29.732							
	2015-01-08,"24,9","19,5","32,8","48,6",0,28.397							
0	2015-01-09,"28,2","21,9",34,"4,4",0,24.886							
1	2015-01-10,"26,76","22,1","34,2",0,1,37.937							
2	2015-01-11,"27,62","22,2","34,8","3,4",1,36.254							
3	2015-01-12,"25,96","21,4","35,4",0,0,25.743							
4	2015-01-13,"25,52","21,2","34,8","0,8",0,26.990							
5	2015-01-14,"25,96","21,3",34,"1,6",0,31.825							
6	2015-01-15,"25,86","21,5","32,8",3,0,25.724							
7	2015-01-16,"26,5","22,3","32,7",2,0,29.938							
8	2015-01-17,"28,86",22,"35,8",0,1,37.690							
9	2015-01-18,"28,26","23,4","35,6",0,1,30.524							
0	2015-01-19,"28,22","22,7","36,5","3,7",0,29.265							
1	2015-01-20,"27,68","23,3","35,6","0,6",0,35.127							
2	2015-01-21,"25,32","22,7","30,9",0,0,29.130							
3	2015-01-22,"21,74","19,2","26,1",31,0,25.795							
4	2015-01-23,"21,04","18,5","26,1","3,6",0,21.784							

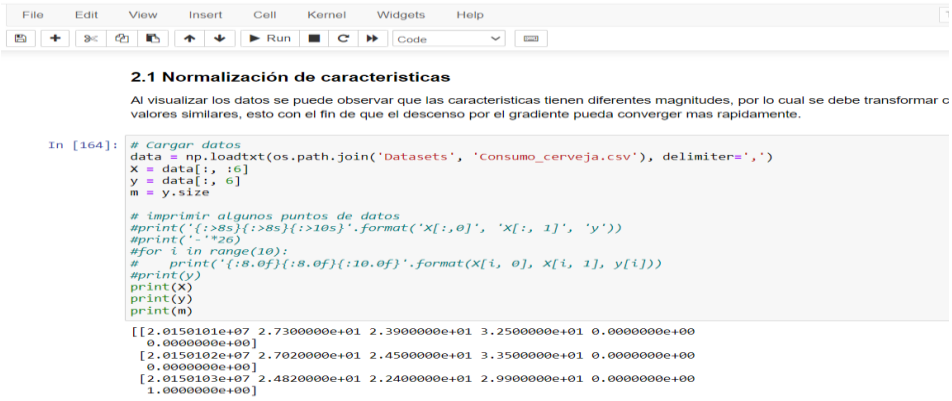
En el Código:

Cargar Dataset de consumo de cerveza, y agregar todas las filas de las columnas 0 al 6 a **X**, y agregar todas las filas de la columna 7 como **Y**, para la predicción de litros de cerveza.

Haciendo uso de Fecha, Temperatura media, Temperatura Mínima, Temperatura Máxima, Precipitación(mm), Fin de semana, como las **X.**

Y consumo de cerveza en Litros como **Y.**

m, para el número de entrenamiento. Con la cantidad de datos que tiene **Y**.



Normalización

```
In [165]: def  featureNormalize(X):
              X_norm = X.copy()
              mu = np.zeros(X.shape[1])
              sigma = np.zeros(X.shape[1])
              print("---")
              print(mu)
              print("---")
              mu = np.mean(X, axis = 0)
              sigma = np.std(X, axis = 0)
              X_norm = (X - mu) / sigma

              return X_norm, mu, sigma
```

```
In [166]: # llama featureNormalize con los datos cargados
          X_norm, mu, sigma = featureNormalize(X)

          print(X)
          print('Media calculada:', mu)
          print('Desviación estandar calculada:', sigma)
          print(X_norm)
```

```
---
[0. 0. 0. 0. 0. 0.]
---
[[2.0150101e+07 2.7300000e+01 2.3900000e+01 3.2500000e+01 0.0000000e+00
  0.0000000e+00]
 [2.0150102e+07 2.7020000e+01 2.4500000e+01 3.3500000e+01 0.0000000e+00
  0.0000000e+00]
 [2.0150103e+07 2.4820000e+01 2.2400000e+01 2.9900000e+01 0.0000000e+00
  1.0000000e+00]
 ...
```

Para normalizar se crea una variable, en este caso es X_**norm** y copia los datos que tenia en la variable X para no alterar su valor original.

Crear en la variable **mu y sigma** arreglos con ceros, del tamaño de lo que es X.

mu, será igual a la media de X en el eje 0(en las columnas), y **sigma** es igual a la desviación estándar de X en el eje 0(en las columnas). Ya preparadas las variables, se hace el calculo de X_**norm** con la formula.

Se Hace la normalización, para que los valores de X estén en una escala similar. Al estar las X en diferentes escalas genera dificultad al aplicar descenso por el gradiente.

Agregar columnas de 1

Despues de `featureNormalize` la funcion es provada, se añade el temino de interseccion a `X_norm`:

```
In [5]: # Añade el termino de interseccion a X
        # (Columna de unos para X0)
        X = np.concatenate([np.ones((m, 1)), X_norm], axis=1)
```

```
In [6]: print(X)
```

```
[[ 1.         -1.64440634  1.9125079  ...  1.365781   -0.41906194
  -0.63124277]
 [ 1.         -1.6415078   1.8243397  ...  1.59772167 -0.41906194
  -0.63124277]
 [ 1.         -1.63860927  1.1315896  ...  0.76273528 -0.41906194
   1.58417656]
 ...
 [ 1.          1.62514101  0.14284628 ... -0.58252056  0.4115282
  -0.63124277]
 [ 1.          1.62803954  0.04838036 ... -0.97681969  0.08896892
  -0.63124277]
 [ 1.          1.63093808  1.11269642 ...  0.55398868 -0.41906194
  -0.63124277]]
```

Coeficientes de Aprendizaje

```
In [403]: # Elegir algun valor para alpha (probar varias alternativas)

alpha = 0.0005 # alpha = 0.003
num_iters = 12900
#alpha = 0.0004 # alpha = 0.003
#num_iters = 90900
# inicializa theta y ejecuta el descenso por el gradiente
theta = np.zeros(7)
theta, J_history = gradientDescentMulti(X, y, theta, alpha, num_iters)

# Grafica la convergencia del costo
pyplot.plot(np.arange(len(J_history)), J_history, lw=2)
pyplot.xlabel('Numero de iteraciones')
pyplot.ylabel('Costo J')

# Muestra los resultados del descenso por el gradiente
print('theta calculado por el descenso por el gradiente: {:s}'.format(str(theta)))

# Estimar la cantidad de litros consumidos de cerveza ,datos 20150111 con una temperatura de "27.62","22.2","34.8",
# y con una precipitacion de 3.4 y fin de semana valor 1.

X_array = [1,20150111,27.62,22.2,34.8,3.4,1]
X_array[1:7] = (X_array[1:7] - mu) / sigma
price = np.dot(X_array, theta)   # Se debe cambiar esto

print('Litros de cerveza Consumido: {:.0f}'.format(price))
```

```
theta calculado por el descenso por el gradiente: [25361.28440654   424.54509511   954.03222143  -183.82907661
  2095.74266551  -768.39526177  2346.05445292]
Litros de cerveza Consumido: 34096
```

Le di el valor de 0.0005 para **Alpha** y 12900 para el numero de interaciones.

Y el valor para **theta** será con 7 ceros, porque es el número de Atributo con el que estoy trabajando en este Dataset.

Datos de prueba para predicción

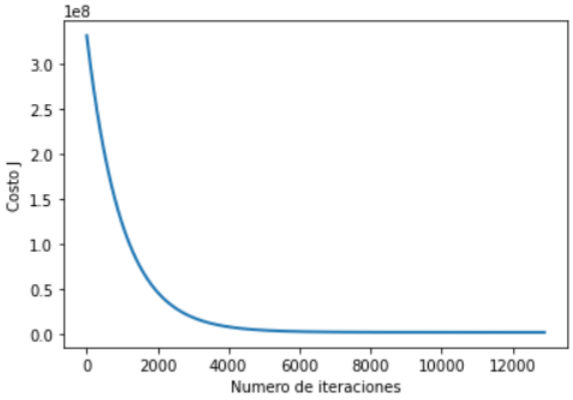
Fecha	Temperatura Media (C)	Temperatura Minima (C)	Temperatura Maxima (C)	Precipitacao (mm)	Final de Semana
20150111	27.62	22.2	34.8	3.4	1

Resultado

```
X_array = [1,20150111,27.62,22.2,34.8,3.4,1]
X_array[1:7] = (X_array[1:7] - mu) / sigma
price = np.dot(X_array, theta)   # Se debe cambiar esto

print('Litros de cerveza Consumido: {:.0f}'.format(price))
```

```
theta calculado por el descenso por el gradiente: [25361.28440654   424.54509511   954.03
  2095.74266551  -768.39526177  2346.05445292]
Litros de cerveza Consumido: 34096
```



In [404]:
```
X_array = [1,20150111,27.62,22.2,34.8,3.4,1]
X_array[1:7] = (X_array[1:7] - mu) / sigma
```

Ecuación de la normal

Cargar Datos y agregar columnas de 1

Utilizando esta formula no requiere que se escale ninguna caracteristica, y se obtendra una solucion exacta con un solo calculo: no hay "bucl convergencia" como en el descenso por el gradiente.

Primero se recargan los datos para garantizar que las variables no esten modificadas. Recordar que no es necesario escalar las caracteristic agregar la columna de unos a la matriz X para tener el termino de intersección(θ_0).

In [12]:
```
# Cargar datos
data = np.loadtxt(os.path.join('Datasets', 'Consumo_cerveja.csv'), delimiter=',')
X = data[:, :6]
y = data[:, 6]
m = y.size
X = np.concatenate([np.ones((m, 1)), X], axis=1)
```

Ecuación de la normal

Una manera de calcular rapidamente el modelo de una regresion lineal es:

$$\theta = (X^T X)^{-1} X^T \vec{y}$$

Utilizando esta formula no requiere que se escale ninguna caracteristica, y se obtendra una solucion exacta con un solo calculo: no hay "bucles de convergencia" como en el descenso por el gradiente.

Primero se recargan los datos para garantizar que las variables no esten modificadas. Recordar que no es necesario escalar las caracteristicas, se debe agregar la columna de unos a la matriz X para tener el termino de intersección(θ_0).

Crear **theta** con el tamaño de las características de X (número de columnas).

Para calcular

X^T usar **X.T**

$(X^T X)^{-1}$ usar **np.linalg.inv(np.dot(X.T,X))**

np.dot para producto de 2

```
In [13]: def normalEqn(X, y):

             theta = np.zeros(X.shape[1])

             theta = np.dot(np.dot(np.linalg.inv(np.dot(X.T,X)),X.T),y)

             return theta
```

Resultado

```
In [423]: # Calcula los parametros con la ecuación de la normal
theta = normalEqn(X, y);

# Muestra los resultados optenidos a partir de la aplicación de la ecuación de la normal
print('Theta calculado a partir de la ecuación de la normal: {:s}'.format(str(theta)));


X_array = [1,20150111,27.62,22.2,34.8,3.4,1]
price = np.dot(X_array, theta)

print('Litros de cerveza: {:.0f}'.format(price))
```

```
Theta calculado a partir de la ecuación de la normal: [-2.49064675e+07  1.23631312e+00 -3.20421098e+01  4.75040702e+01
  6.76630650e+02 -5.84580096e+01  5.19947694e+03]
Litros de cerveza: 34096
```

Resultado de la regresión lineal

Se realizo pruebas con diferentes valores para Alpha y números de iteraciones, daban resultados, aproximados, pero se encontró valores que sacan el mismo resultado que la ecuación de la normal, Así como:

para **Alpha** = 0.0005 y **num_iters**= 12900, Da un resultado de **34096.**

Resultado de ecuación de la normal

El resultado que da es **34096.**