Proyecto 1

Regresión lineal (recta) simple (de una variable)

Descripción

El caso más simple de regresión es cuando los datos se aproximan por medio de una recta y se busca una relación sólo entre dos variables, una independiente **x** y una dependiente **y**.
Hacer un programa en Python que reciba como entrada un conjunto de ejemplos de la forma **(entrada,salida)** y regrese los valores de Theta **(θ_0 y θ_1)** que mejor ajusten al conjunto de datos. La recta debe ser calculada por el algoritmo de **regresión lineal**.

Datos

Los datos están en un archivo (ex1data1.txt) de 2 columnas. Éstos se utilizan para predecir las ganancias de un *food truck*. Suponga que lo contrató el CEO de un restaurante que tiene varios carros de comida en diferentes ciudades. El CEO está considerando poner un carro de comida en una nueva ciudad para lo cual cuanta con los datos de sus otros carros (los que vienen en el archivo) sobre el tamaño de la población (primera columna) y las ganancias del carro de la ciudad (segunda columna), dados en 10,000s. El archivo deberá ser separado en dos vectores:

1. El vector **X** conteniendo TODAS las entradas. Se refieren a la población.
2. El vector **y** conteniendo TODAS las salidas. Se refieren a las ganancias.

Su tarea es hacer una herramienta que le permita seleccionar la mejor ciudad para poner un nuevo *food truck*.

Fuente: A. Ng. Machine Learning. Curso de Coursera-Stanford (2011).

Notas

- Recuerde que para facilitar la vectoriazación es necesario agregar una columna de 1´s a la matriz de entradas X.
- Para poder vectorizar tiene que contar con operaciones con matrices por lo que puede usar una librería que las proporcione.

Funciones

Se deberán programar 3 funciones

1. **`graficaDatos(X,y,theta)`**. Recibe los datos de entrada y las grafica como puntos en un plano **(x,y)**. Además, recibe un vector theta y grafica sobre los datos la recta que resulte de esos valores.

2. **`gradienteDescendente(X,y,theta,alpha,iteraciones)`**. Recibe:
    a. Datos de entrada ya separados en vectores **X** y **y**.

b. Vector **theta** = $[\theta_0,\theta_1]$ inicial. Dicho vector puede estar inicializado en 0 para ambos valores.
c. Razón de aprendizaje **alpha**. Se recomienda probarlo con un valor **alpha** = 0.01.
d. El número de iteraciones **iteraciones**. El número de iteraciones que va a realizar el algoritmo. Se recomienda probarlo con iteraciones = 1500.
e. Se puede hacer *batch* u *on-line* (los resultados que se dan para prueba se obtuvieron con *batch* pero con *on-line* deben ser muy cercanos).

La función deber regresar el valor del vector **theta** final.

3. **`calculaCosto(X,y,theta)`**. Recibe las entradas **X** y **y**, y un vector **theta** y regresa el valor de la función de costo **$J(\theta_0,\theta_1)$** que resulta.

Resultados de prueba

Una vez entrenado el sistema, es decir, una vez encontrados los valores de los parámetros para la recta de regresión, se pueden hacer las siguientes pruebas:

Preducion1 = [1, 3.5]*theta, lo cual debe regresar un valor de 0.4519767868
Prediccion2 = [1,7]*theta, lo cual debe regresar un valor de 4.5342450129

Recuerde que $\boldsymbol{h_\theta(x_i) = \theta^T X^T = X\theta = \theta_0 + \theta_1 x_i}$ donde **X** representa en este caso el vector de ejemplos con la columna de 1 agregada. Por esta razón, las predicciones se obtienen multiplicando una entrada **[1,x]** por el vector theta (que es un vector columna).

Pueden también imprimir el costo al inicio, antes de llamar a **`gradienteDescendente`**, con el vector inicial de theta igual a **[0,0]**. Luego lo imprimen después de correr la función y el vector debe disminuir casi hasta 0.

Vectorización

VECTORICE TODAS las operaciones que pueda vectorizar, utilizando alguna librería para manejo de matrices.

Entrega y Revisión

- Salve su archivo con el nombre **Proyecto1.py**. Tiene que ser con ese nombre porque es el que se va a usar para la revisión automática.
- Suba sus archivos dos archivos, el del proyecto (.py) y el de la documentación (.docx o .pdf) **por separado**, NO los ponga en ZIP.
- Para la revisión se harán pruebas sobre las funciones solicitadas. Para esto, tanto el archivo que contiene las funciones (el suyo) como el de prueba (el mío), se colocarán en el mismo folder. El programa de prueba al inicio hará un:

```
from file import function
```

y luego se utilizará en el script de prueba llamándola como como:

```
function(<parámetros>)
```