Práctica 1

Regresión lineal, función de costos y gradiente descendente

En esta práctica deberá implementar el algoritmo completo de regresión lineal, haciendo uso de la función de costos y el algoritmo de gradiente descendente visto en clase. Se te dará un set de datos para que entrenes tu algoritmo. Una vez que se haya entrenado, deberá entregar los valores ϴ que caracterizan a la hipótesis encontrada, adicionalmente deberá graficar dicha función.

Regresión lineal

Usando el archivo ex1data1.txt deberá graficar sus valores para poder observar su distribución y familiarizarse con los datos. Una vez que haya cargado este archivo al entorno de Octave deberá crear dos variables una *"X"* donde se guardará la primer dimensión de la matriz que contiene los datos del archivo y otra variable *"y"* donde estarán los datos de la segunda dimensión.

```
data = load('ex1data1.txt');
X = data(:, 1); y = data(:, 2);
m = length(y);

plot(x, y, 'rx', 'MarkerSize', 10);
ylabel('Profit in $10,000s');
xlabel('Population of City in 10,000s');
```

Una vez hecho esto se deberá de desplegar la siguiente gráfica.

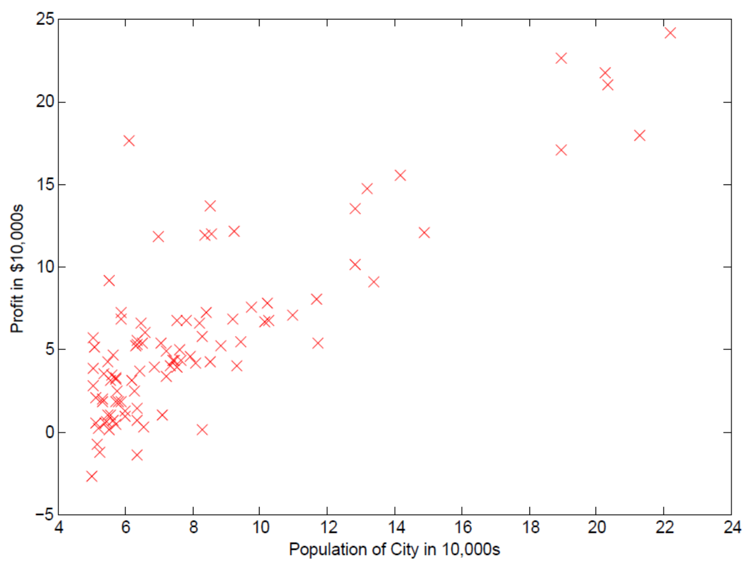

Ilustración 1. Gráfica de set de datos para entrenamiento

Gradiente Descendente y Función de costos

Como se vió en clase, el objetivo de la regresión lineal es minimizar la función de costos

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Donde la hipótesis $h_\theta(x)$ está dada por el siguiente modelo lineal

$$h_\theta(x) = \theta^T x = \theta_0 + \theta_1 x_1$$

Recuerde que los parámetros de este modelo son los θ_j . Al ajustar estos valores usted estará minimizando la función de costo $J(\theta)$. En este caso se usará el algoritmo de gradiente descendente en donde en cada iteración se ajustan estos parámetros con el uso de la siguiente ecuación.

$$\theta_j := \theta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}$$

Recuerde que todos los parámetros θ deben de actualizarse al mismo tiempo, de esta manera con cada iteración los valores de estos parámetros lo acercarán al mínimo de la función de costo.

Nota de implementación*: Tome en cuenta que estamos tratando a todo como matrices y vectores, de tal manera que use las operaciones de álgebra lineal para facilitar la implementación de las ecuaciones con las operaciones de matrices vistas en clase.*

Cálculo de costo

Para saber si el algoritmo converge es necesario saber si la función de costos $J(\theta)$ converge, de tal manera que es necesario monitorearla para confirmar. En el archivo computeCost.m , tiene que completar el código para calcular el valor de dicha función a partir de los valores del set de entrenamiento, recuerde que estos valores están almacenados en *"X"* y *"y"* y no son escalares, sino mas bien matrices.

Gradiente descendente

En el archivo gradientDescent.m se ha escrito la estructura del loop que se necesita para dicha función, sin embargo usted debe de completar el código para poder calcular los nuevos valores de θ , mismos que se actualizarán por cada iteración del programa. Use las ecuaciones que se vieron en clase para poder hacer el computo de los valores, adicionalmente usted puede observar el valor de la función de costos con cada iteración para verificar que este valor dismunuye envez de aumentar, de tal manera que si el valor de la función decrece su implementación será correcta. Note como esta función llama a la anterior para calcular la función de costo. Los valores finales de θ serán los que use para graficar la hipótesis final, refiera se ala ilustración 2.

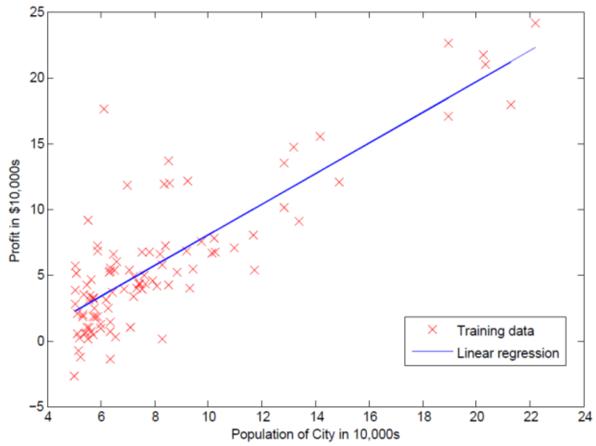

Ilustración 2. Set de entrenamiento con hipótesis final obtenida a partir del algoritmo.