Regresión Lineal Multidimensional Masiva para BigData. Teoría Básica y Trabajo Práctico.

1. Introducción

BigData es una denominación que se ha popularizado en relación al análisis y extracción de conocimiento útil contenido en grandes volúmenes de datos. Todos los grandes problemas involucrados en BigData, especialmente los concernientes al análisis de datos, han estado presentes en diferentes disciplinas desde hace décadas. Lo que ha emergido es la posibilidad práctica asequible de procesamiento y análisis de grandes volúmenes, más allá de la posibilidad teórica que siempre ha sido considerada.

Conforme el volumen de datos aumenta en una tarea de análisis, se incrementan los costes computacionales en forma no-lineal y en muchos casos de forma exponencial. En muchas situaciones es preciso centrarse en procedimientos relativamente sencillos en cuanto a coste computacional para mantener acotados los costes. Ejemplo de estos procedimientos son aquellos que mantienen modelos de costes computacionales de orden polinómico en en relación a la dimensión del problema. Se recomienda en este punto recordar los criterios de clasificación de los algoritmos en relación a su complejidad.

Uno de los ejemplos de procedimientos de análisis de datos con coste contenido polinómico es el de análisis o ajuste de Regresión, ampliamente utilizado en su versión unidimensional, pero del que presentaremos y trabajaremos en su versión multidimensional.

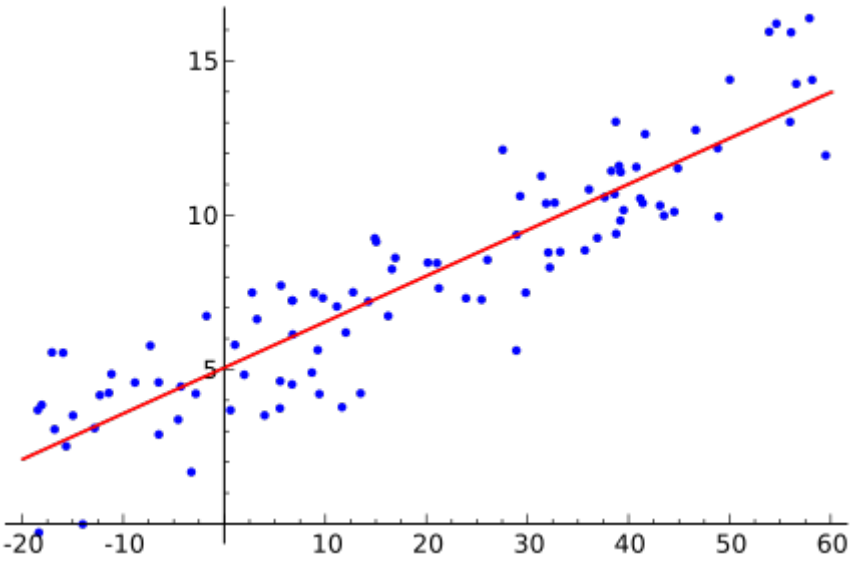
2. Regresión Lineal Multidimensional

En la Regresión Lineal Multidimensional tratamos de encontrar un ajuste de datos para que una determinada función pueda representarse mediante una función lineal, pero de un conjunto de datos con muchas dimensiones.

$$z = A_1 x_1 + \cdots + A_n x_n + A_{n+1} = A_{n+1} + \sum_{i=1}^{n} A_i x_i$$

Donde la función que deseamos representar es z que depende de un conjunto de variables independientes $x_i, i = 1, \dots, n$. Para ello, es necesario estimar los coeficientes o parámetros $A_i, i = 1, \dots, n+1$.

Generalmente, la forma más utilizada de Regresión es la unidimensional, en donde solamente se involucra una variable. En este caso la formula anterior se corresponde con una recta de regresión.



El caso de una dimensión ha sido muy utilizado en Estadística a un nivel sencillo, pero para estudios más realista como los involucrados en el análisis de volúmenes considerables de datos, como en BigData, es más adecuado considerar la regresión multidimensional en donde se involucran múltiples variables.

Para homogeneizar la representación de la formula anteriormente presentada, y dado que el parámetro A_{n+1} aparece de una forma singular, se suele incluir una representación aumentada o ampliada en la que en lugar de trabajar con las coordenadas x originales se introducen las coordenadas ampliadas y con un 1 adicional en la dimensión $n+1$, en la forma siguiente:

$$(y_1, y_2, \dots, y_n, y_{n+1}) = (x_1, x_2, \dots, x_n, 1)$$

En cuyo caso la representación gana en simplicidad:

$$z = \sum_{i=1}^{n+1} A_i y_i$$

3. Estimación de Coeficientes. Modelo de componentes

Realmente, los que se dispone es de un conjunto de m muestras de valores de la función y de las coordenadas multidimensionales. A partir de esas muestras se pretende estimar los coeficientes A_i

El algoritmo utilizado generalmente consiste en formular un modelo de error obtenido para el conjunto de muestras dependiente de los coeficientes y tratar de minimizar ese error. El modelo de Error Mínimo Cuadrático (Minimum Mean Square Error) es el siguiente:

$$Min\ E^2(A_1, \dots, A_{n+1}) = \sum_{j=1}^{m} \left(z_j - \sum_{i=1}^{n+1} A_i y_{ij} \right)^2$$

Si el error se pretende que sea mínimo, se debe verificar que:

$$\frac{\partial E^2}{\partial A_k} = 0 \qquad k = 1, \dots, n+1$$

Eso conduce a que se deba verificar que:

$$\sum_{j=1}^{m} y_{kj} \left(z_j - \sum_{i=1}^{n+1} A_i y_{ij} \right) = 0 \qquad k = 1, \dots, n+1$$

Lo que se puede expresar de forma más compacta como:

$$\sum_{j=1}^{m} y_{kj} z_j = \sum_{i=1}^{n+1} \left(\sum_{j=1}^{m} y_{kj}\, y_{ij} \right) A_i \qquad k = 1, \dots, n+1$$

Que constituye un sistema lineal con $n+1$ ecuaciones e igual número de incógnitas.

4. Estimación de Coeficientes. Modelo Matricial

Para simplificar la programación de los procedimientos de cálculo, es más adecuado trabajar con un modelo matricial. Sean las siguientes matrices y vectores obtenidos a partir de los datos. El vector de m filas con las muestras de la función:

$$Z = \begin{pmatrix} z_1 \\ \vdots \\ z_m \end{pmatrix}$$

La matriz de m filas y n columnas con las muestras de las variables y su representación aumentada con n+1 columnas:

$$X = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix}$$

$$Y = \begin{pmatrix} x_{11} & \cdots & x_{1n} & 1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{m1} & \cdots & x_{mn} & 1 \end{pmatrix}$$

El vector de n+1 filas con los coeficientes incógnitas:

$$A = \begin{pmatrix} A_1 \\ \vdots \\ A_{n+1} \end{pmatrix}$$

En este caso el error se puede representar por:

$$D = Z - YA$$

$$E^2 = D^T D$$

Donde el vector D es el error cometido en cada muestra. La estimación de los coeficientes por el siguiente sistema lineal:

$$(Y^T Y)A = Y^T Z$$

Cuya solución se obtiene como:

$$A = (Y^T Y)^{-1} Y^T Z$$

Pero no es necesario obtener A de esta última forma, sino mediante la resolución del sistema lineal.

NOTA: recuerden que m debe ser muy superior a n.

5. Tareas Computacionales

Las Tareas que realizaremos en el Trabajo son las siguientes:

1. Generar en MATLAB conjuntos de datos a partir de la definición de los números de muestras m y la dimensión del espacio n. El programa generador y la estructura de los datos que se escribirán en un fichero se definirá en el Anexo.
2. Construir un Programa C/C++ que inicialmente leerá un conjunto de datos desde un fichero definido.
3. Construir utilizando MKL (se utilizarán ambas librerías BLAS y LAPACK). Incluir en el programa que estime el vector de coeficientes y el error.
4. Variar las dimensiones del fichero de datos y evaluar las prestaciones dentro de los limites de funcionamiento.
5. Realizar una variante de el procedimiento anterior pero utilizando GPU y programación CUDA (se utilizarán las librerías cuBLAS y cuSolver denso). Minimizar las transferencias de datos. Repetir el análisis de prestaciones.

Anexo. Funciones generadoras de datos en MATLAB.

1. Para generar conjuntos de datos utilizaremos la función MATLAB DatosRegresion()

[A] = DatosRegresión(m,n,filename,E);

Proporciona el valor exacto del vector de coeficientes utilizado en la generación de los datos. Los argumentos necesarios son:

m: número de muestras a generar. Debe ser muy superior al de dimensión n.

n: dimensión del espacio de las variables.

filename: el nombre del fichero donde se guardarán los datos, y desde donde se leerán en la fase de estimación. Utilizar extensión .txt para poderlo leer de forma sencilla.

E: un parámetro de error. Por ejemplo 0.1 para un 10% de nivel de ruido.

2. También se incorpora la función EstimaRegresion() que realiza una estima de forma análoga a como deberá realizarla el programa C/C++

[A,E] = EstimaRegresión(filename);

El argumento filename es el nombre del fichero de datos. Los resultados son una estima del vector de coeficientes y del nivel de ruido o error. Evidentemente los resultados pueden diferir razonablemente de los datos exactos.

3. La estructura del fichero de datos se compone de diversas filas:

Primera fila: el número entero con la dimensión n.

Segunda fila: el número de muestras m

Filas siguientes (tantas filas como m): En cada fila, el primer dato o columna es el valor de z y las n columnas siguientes los valores de las coordenadas x de esa muestra.