Proyecto 3 Inteligencia Artificial

Saldaña Hernandez Norman Brandon Guerrero Ferrusca Francisco

González Lorenzo Daniela Michel Tenorio Veloz Alisson Dafne

Noviembre 2021

1 Planteamiento del Problema

Implementar un algoritmo de regresión Lineal generalizada con las siguientes características:

1. Implementar una función para calcular el polinomio resultante de la diferencia de cuadrados
2. Implementar una función para calcular la derivada parcial de un polinomio al cuadrado
3. Implementa un algoritmo que encuentre la solución de un sistema de ecuaciones

2 Definición del problema

El problema es la implementación de un algoritmo de un proceso conocido como regresión lineal generalizada, este proceso se puede representar por medio de la siguiente ecuación:

$$\mathbf{S}(\beta_j) : \sum_{i=1}^{n}(y_i - (\beta_0 + \sum_{j=1}^{m}\beta_j x_{i,j}))^2 \tag{1}$$

La regresión Lineal para n variables consiste minimizar la función $S(\beta_j)$ es decir, que se cumpla:

$$\frac{dS}{d\beta_j} = 0 \tag{2}$$

al final debemos obtener una serie de valores β para construir la siguiente ecuación

$$y = \beta_0 + \sum_{j=1}^{m}\beta_j X_j \tag{3}$$

De esta forma podemos hacer predicciones para cuando tengamos un nuevo arreglo de X poder saber que Y correspondería.
Nótese que j + 1 es la cantidad de valores β van que obtendremos, es decir desde β_0 hasta β_m, β_0 es un parámetro que sirve de ofset del resultado y.

3 Solución

La solución se llevó a cabo en Matlab ya que es un software que permite programación en alto nivel orientado a manejo de datos, sobre todo de matrices, en este caso la sintaxis resulta sencilla y el manejo de arreglos de datos no requiere un manejo de memoria implicito. El software está optimizado cuando se usan solo matrices.

3.1 Polimonio al cuadrado

Para entrar de lleno a la resolución primero hay que plantear de qué manera vamos a identificar las variables β, también pensamos en una resolución de un polinomio al cuadrado y la derivada de tales polinomios.Como pensamiento inicial implementamos el código usando variables simbolicas, funciones como simplify y diff, sin embargo al ya estar implementadas estas funciones en Matlab resultó no ser una resolución adecuada con los requerimentos del proyecto.
Replanteando el problema y enfocandonos a la especialidad de Matlab (matrices) empezamos por el siguiente planteamiento. Inicialmente tenemos el siguiente dataset que se puede ver como vectores columna de datos, como se muestra en (4)

$$\begin{bmatrix} \mathbf{y} \\ 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \begin{bmatrix} \mathbf{x_1} \\ 6 \\ 5 \\ 7 \\ 10 \end{bmatrix} \begin{bmatrix} \mathbf{x_2} \\ 16 \\ 5 \\ 31 \\ 9 \end{bmatrix} \cdots \begin{bmatrix} \mathbf{x_{m-1}} \\ 2 \\ 5 \\ 1 \\ 29 \end{bmatrix} \begin{bmatrix} \mathbf{x_m} \\ 15 \\ 7 \\ 3 \\ 1 \end{bmatrix} \tag{4}$$

La idea es que de esta manera acomodemos estos vectores columna para que tengamos una representación de la ecuación (5) sin tener que usar variables simbólicas.

$$(y_i - (\beta_0 + \sum_{j=1}^{m} \beta_j x_{i,j})) \tag{5}$$

Como vemos en la ecuación 5, nos damos cuenta que podríamos poner cada termino en un vector que llamaremos **p**, donde podemos relacionar cada término de la ecuación con una columna.
Tomando como ejemplo (4), el vector **y** tiene como primer término en número 1, ese sería nuestra $\mathbf{y_i}$ cuando i vale 1, es decir, $\mathbf{y_1}$, luego para β_0 el coeficiente de este término siempre es 1, y como nos damos cuenta el factor de cada término β_j debe multiplicarse por $x_{i,j}$ por lo tanto tenemos:

$$p = \begin{bmatrix} -\mathbf{y} & \beta_0 & \beta_1 & \beta_2 & \cdots & \beta_{j-1} & \beta_j \\ y_1 & 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,j-1} & x_{1,j} \end{bmatrix} = \begin{bmatrix} -\mathbf{y} & \beta_0 & \beta_1 & \beta_2 & \cdots & \beta_{j-1} & \beta_j \\ 1 & 1 & 6 & 16 & \cdots & 2 & 15 \end{bmatrix} \tag{6}$$

(6) nos dá el vector **p** para la primera fila de (4), pero hasta ahora esto solo representa un polinomio(el de la primera fila) , pero lo que en realidad necesitamos es algo como la ecuación (7) que es parte de la ecuación original de regresión lineal, la diferencia es que en (7) tenemos que hacer la suma de todos los vectores **p**

$$\sum_{i=1}^{n} (y_i - (\beta_0 + \sum_{j=1}^{m} \beta_j x_{i,j})) \tag{7}$$

Para eso en realidad nos dimos cuenta que aún no es necesario sumar, dado que lo que debemos sumar son los polinomios elevados al cuadrado y hasta ahora aun no hacemos eso, sin embargo podemos tener una matriz que contenga a todos los vectores p,por ahora solo nos interesa tener todos los vectores **p**. La matriz de vectores **p** quedaría como sigue:

$$\begin{pmatrix} -y_{1,1} & 1 & x_{1,1} & \cdots & x_{n-1,m-1} & x_{n,m} \\ -y_{2,1} & 1 & x_{2,1} & \cdots & x_{n-1,m-1} & x_{n,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -y_{n-1,1} & 1 & x_{n-1,1} & \cdots & x_{n-1,m-1} & x_{n,m} \\ -y_{n,1} & 1 & x_{n,1} & \cdots & x_{n-1,m-1} & x_{n,m} \end{pmatrix} \tag{8}$$

A manera de ejemplo, para un vector **y** y un vector **x**, la composición de vectores p se vería como a continuación, siendo esta una aplicación de (8) para un problema de regresión con 1 variable x, donde la primera columan representa a y, la segunda al coeficiente de β y la tercera al coeficiente de β_1, como ya se habia dicho antes.

$$\begin{pmatrix} y & x \\ 1 & 6 \\ 2 & 5 \\ 3 & 7 \\ 4 & 10 \end{pmatrix} \rightarrow \begin{pmatrix} y & \beta_0 & \beta_1 \\ -1 & 1 & 6 \\ -2 & 1 & 5 \\ -3 & 1 & 7 \\ -4 & 1 & 10 \end{pmatrix} \tag{9}$$

Ahora bien, cada vector P debe elevarse al cuadrado, para lograr esto nos preguntamos la definición de un polinomio elevado al cuadrado, el proceso que se lleva a cabo es que cada elemento del polinomio se multiplica por cada elemento de ese mismo polinomio, esto al verlo en forma de matrices es sumamente semejante a multiplicar un vector fila por su transpuesta, es decir un vector columna.
A continuación llamaremos a cada elemento de p como a_m (10), solo para ver el proceso de multiplicación, esto sirve para ver mas facilmente que información nos dá la matriz resultante.)

$$p = \begin{bmatrix} a_1 & a_2 & \cdots & a_{m-1} & a_m \end{bmatrix} \tag{10}$$

A manera de ejemplo supongamos que **p** solo tiene 3 elementos (11).

$$p = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \tag{11}$$

Entonces la multiplicación de **p** por su transpuesta $\mathbf{p}^T$ sería nuestro elemento de la suma que se representa en (7), elevada el cuadrado. Donde cada elemento de la matriz contiene cada una de las multiplicaciones de los términos del polinomio.

$$p^2 = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{pmatrix} a_1a_1 & a_1a_2 & a_1a_3 \\ a_2a_1 & a_2a_2 & a_2a_3 \\ a_3a_1 & a_3a_2 & a_3a_3 \end{pmatrix} = M \tag{12}$$

Ahora tomemos en cuenta que cada elemento de p tiene la información de la ecuación (5), organizada como se ve a continuación:

$$p_\beta^2 = \begin{bmatrix} y & \beta_0 & \beta_1 \end{bmatrix} \begin{bmatrix} y \\ \beta_0 \\ \beta_1 \end{bmatrix} = \begin{pmatrix} yy & y\beta_0 & y\beta_1 \\ \beta_0 y & \beta_0\beta_0 & \beta_0\beta_1 \\ \beta_1 y & \beta_1\beta_0 & \beta_1\beta_1 \end{pmatrix} \tag{13}$$

De manera abstracta llamaremos M al cuadrado a nuestro vector p multiplicado por su transpuesta:

$$Mi = p_i^2 = p_i * p_i^T \tag{14}$$

De (13) podemos obtener conclusiones interesantes, consideremos un ejemplo, para el siguiete conjunto de datos vamos a obtener $\mathbf{p}_1^2$,

$$\begin{pmatrix} y & x \\ 1 & 6 \\ 2 & 5 \\ 3 & 7 \\ 4 & 10 \end{pmatrix} \rightarrow p_1 = \begin{bmatrix} 1 & 1 & 6 \end{bmatrix} \rightarrow p^2 = \begin{bmatrix} -1 & 1 & 6 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 6 \end{bmatrix} = \begin{pmatrix} (1)(1) & (1)(1) & (1)(6) \\ (1)(1) & (1)(1) & (1)(6) \\ (6)(1) & (6)(1) & (6)(6) \end{pmatrix} = \begin{pmatrix} 1 & -1 & -6 \\ -1 & 1 & 6 \\ -6 & 6 & 36 \end{pmatrix} \tag{15}$$

Ahora comparemos con como sería el desarrollo del polinomio al cuadrado sin usar matrices:

$$S_1 = (y_1 - (\beta_0 + \beta_1 x_{1,1}))^2 = (1 - (\beta_0 + 6\beta_1))^2 = (-1 + \beta_0 + \beta_1)^2 = \tag{16}$$

$$= 1 - \beta_0 - 6\beta_1 - \beta_0 + \beta_0^2 + 6\beta_0\beta_1 - 6\beta_1 + 6\beta 1\beta_0 + 36\beta_1^2 \tag{17}$$

$$= 1 - 2\beta_0 - 12\beta_1 + \beta_0^2 + 12\beta_0\beta_1 + 36\beta_1^2 \tag{18}$$

Lo interesante es como pasamos de (16) a (17), basicamente es una simplificación, agrupamos los términos con términos comunes y sumamos sus coeficientes, tambien podemos notar que (16) es practicamente lo mismo que la matriz de (14), de aquí tenemos las siguientes conclusiones:

1. Los elementos de la diagonal principal contienen información de los coeficientes del polinomio resultante que estan elevados al cuadrados, en este ejemplo la diagonal principal de (14) es [1 1 36], que si comparamos con (17) son los coeficientes de y^2, β_0^2 y β_1^2

2. El resto de coeficientes viene dado por M(i,j) + M(j,i), esto es interesante porque podemos ver por ejemplo el termino $y\beta 0$, este término lo podemos localizar en (13) y vemos que se repite en la posición M(1,2) y M(2,1), de manera similar para el término β_1 sus coeficientes aparecerían en M(1,3) y M(3,1), y para los términos compuestos que en este caso sería el término $\beta_0\beta_1$ sus coeficientes aparecen en M(2,3) y M(3,2), por lo tanto podemos decir que para cada relación de fila columna el coficiente de la expresión elevada al cuadrado simplificada (18) se puede obtener de sumar los elementos de la matriz M que correspondan a M(i,j) y M(j,i).
   Para obtener (18) a partir de (15) sería:

$$M(1,1) + (M(1,2) + M(2,1))\beta_0 + (M(1,3) + M(3,1))\beta_1 + (M(2,3) + M(3,2))\beta_0\beta_1 + M(2,2)\beta_0^2 + M(3,3)\beta_1^2 \tag{19}$$

sustituyendo tenemos:

$$1+(-1-1)\beta_0+(-6-6)\beta_1+(6+6)\beta_0\beta_1+1\beta_0^2+36\beta_1^2 \tag{20}$$

y simplificando:

$$1-2\beta_0-12\beta_1+12\beta_0\beta_1+1\beta_0^2+36\beta_1^2 \tag{21}$$

y como vemos (21) es igual a (18), por lo tanto podemos concluir una relación entre filas y columnas que indican que coeficientes tendrán los términos de la expresión que buscamos obtener, nótese que si tuvieramos más términos β podemos seguir este mismo proceso, por ejemplo si nos sale un término $\beta_0\beta_2$ podemos encontrar sus coeficientes como la suma de M(2,4) y M(4,2) ya que β_0 está relacionado con la columna 2 y β_2 con la columa 4.

3. Otra conclusión es que la matriz M es una matriz simétrica, dado que los elementos de la triangular superior son iguales a los de la matriz triangular inferior, esto solo simplifica nuestro trabajo ya que para obtener los coeficientes de un término no debemos buscar M(i,j) y M(j,i), simplemente basta con uno de ellos multiplicarlo por 2 ya que el término se repite.

Ahora bien, hemos obtenido la definición de matriz M y como esta representa un polinomio al cuadrado (Planteamiento del problema 1), sin embargo si regresamos a la ecuación (1), requerimos n matrices **M**, estas obtenidas de cada vector **p**, por lo tanto para tener una representación de la ecuación 1 basta hacer lo siguiente:

$$S=\begin{bmatrix}y_{1,1} & 1 & x_{1,1} & \cdots & x_{n-1,m-1} & x_{n,m}\end{bmatrix}^2+\begin{bmatrix}y_{2,1} & 1 & x_{2,1} & \cdots & x_{n-1,m-1} & x_{n,m}\end{bmatrix}^2+\cdots \tag{22}$$

$$+\begin{bmatrix}y_{n-1,1} & 1 & x_{n-1,1} & \cdots & x_{n-1,m-1} & x_{n,m}\end{bmatrix}+\begin{bmatrix}y_{n,1} & 1 & x_{n,1} & \cdots & x_{n,m-1} & x_{n,m}\end{bmatrix} \tag{23}$$

O lo que es igual

$$S=M_1+M_2+M_3+\cdots+M_{i-1}+M_i \tag{24}$$

3.2 Derivadas parciales de un polinomio al cuadrado

Ya que todas las matrices M son del mismo tamaño se puede hacer la suma elemento a elemento sin ningun problema, de hecho como se puede notar en ningun momento hemos hecho la simplificación de la expresión resultante de cada matriz M, esto es porque aun no es necesario, en este punto una vez obtenida la Matriz **S** (24) ya podríamos simplificar como se explicaba anteriormente. Sin embargo aun no es necesario, dado que lo que en realidad queremos es la derivada parcial de S con respecto a cada β_j para igualarla a cero y así minimizar la función, como marca la ecuación (2).
¿Esto como se hace?, nos dimos cuenta que la matriz S tiene la misma forma y lógica que las matrices M específicas para cada vector p, ya que es una suma de ellas, entonces tomamos el siguiente concepto de derivada parcial:
Una derivada parcial solo toma en cuenta aquellas expresiones matemáticas que contengan a la variable en cuestion con la que se va a derivar al respecto, por ejemplo, si tenemos la expresión $17\beta_0\beta 2$ y deseamos derivar con respecto a β_1 la derivada parcial será 0, porque no está contenida la variable en cuestión, pero si tenemos el caso que queremos derivar $8\beta_0\beta_1$ nos quedará como resultado $8\beta_0$. Retomemos (13) y reescribamosla a continuación:

$$\begin{pmatrix} yy & y\beta_0 & y\beta_1 \\ \beta_0 y & \beta_0\beta_0 & \beta_0\beta_1 \\ \beta_1 y & \beta_1\beta_0 & \beta_1\beta_1 \end{pmatrix} \tag{25}$$

Notemos algo interesante:

1. Si queremos derivar con respecto a β_0 podemos notar que tanto la columna 2 como la fila 2, ambas contienen a β_0 en cada uno de sus coeficientes, y practicamente la columna 2 y la fila 2 son igual por la propiedad del conmutabilidad en el operador multiplicación entonces para derivar parcialmente es conveniente no simplificar, sino tomar directamente una fila o columna relacionada con la variable con la que queremos derivar al respecto.

2. Del punto anterior podemos darnos cuenta que la derivada parcial es la fila o columna en cuestion multiplicada por 2, por ejemplo la derivada parcial de S con respecto a β_0 es la columna 2 multiplicada por 2 (multiplicación de un escalar por un vector), ¿porque?, porque como se dijo, el coeficiente se conserva, y como sabemos del análisis de M, cada element debe multiplicarse por 2 para obtener el coeficiente relacionado, esto aplica solo para elementos que no son la diagonal principal, pero en este caso por mera coincidencia resulta que el elemento de la matriz principal contiene a la variable elevada al cuadrado por lo que su derivada es 2 veces la variables (p.e $\frac{d}{d\beta_0}\beta_0^2=2\beta_0$), entonces ya sea que tomemos filas o columnas de la matriz S, al multiplicarlas por 2 obtendremos que casa una de ellas es la derivada parcial con respecto a la variable relacionada a su respectiva columna, exceptuando la primera columna ya que la primera columna es de términos y que son idependientes de β_j

De lo anterior podemos obtener una matriz **dS**, para ejemplificar, a continuación escribiremos la matriz **S** resultante del planteamiento (9), en este caso dejaremos el cálculo de M_2, M_3 y M_4 al lector, usando el mismo proceso que usamos para obtener M_1

$$S=M_1+M_2+M_3+M_4=\begin{bmatrix} 30 & -10 & -77 \\ -10 & 4 & 28 \\ -77 & 28 & 210 \end{bmatrix} \tag{26}$$

Entonces la derivada parcial de **S** con respecto a β_0 sería 2*[-10 4 28] =[-20 8 28], donde asignaremos a cada elemento de este vector obtenido su respectiva variable y nos quedaría: $-20 + 8\beta_0 + 28\beta 1$
Para la derivada de **S** con respecto a β_1 quedaría: $-144 + 56\beta_0 + 420\beta_1$ y a partir de esto podemos construir una matriz **dS** que contenga todas las derivadas parciales, solo bastaría con multiplicar por 2 la matriz S y quitar la primera fila.
Para este caso **dS** quedaría así:

$$dS = \begin{bmatrix} -20 & 8 & 28 \\ -144 & 56 & 420 \end{bmatrix} \tag{27}$$

de forma generalizada tendríamos que dS sería:

$$dS = 2 \begin{bmatrix} S(2,:) \\ S(3,:) \\ \vdots \\ S(r-1,:) \\ S(r,:) \end{bmatrix} \tag{28}$$

donde los dos puntos representar todos los elementos de las columnas y r representa el tamaño de S que debe ser el numero de variables(m) mas 2 unidades ya que se agrega β_0 y y como parte del vector **p** que al elevarse al cuadrado resulta en una matriz cuadrada de m+2 x m+2, entonces r es igual a m+2, y la matriz **dS** tiene tamaño m+1 x m+2, si tenemos 1 variable entonces la matriz S sería de 3x3 como ya hemos visto en (26) y la matriz dS será de 2x3 como vimos en (28).

3.3 Resolución de un sistema de ecuaciones

Por ultimo, dS podemos separarla en un vector columa (el primero que contiene los términos independientes) y una matriz cuadrada (el resto de la matriz que contiene los coeficientes de las expresiones de derivadas parciales)
por ejemplo para el ejemplo de (27) podríamos verlo de la siguiente forma:

$$dS \xrightarrow[descomponiendo]{} B = \begin{bmatrix} -20 \\ -144 \end{bmatrix} A = \begin{bmatrix} 8\beta_0 & 28\beta_1 \\ 56\beta_0 & 420\beta_1 \end{bmatrix} \tag{29}$$

Donde el vector de términos independientes podemos llamarlo B y la matriz de coeficientes la llamaremos A
Un método muy útil para la resolución de sistemas de ecuaciones es obtener un vector C que contendría las soluciones de las variables β_j que buscamos y este vector C se obtiene como:

$$C = inv(A) * B \tag{30}$$

Donde inv(A) es la matriz inversa de A. Así se resolvería el problema de regresión lineal y así obtendíamos los valores β para obtener nuestro modelo buscado que está descrito en la ecuación (3).

4 Desarrollo del código

4.1 Polimonio al cuadrado

para su obtención primero creamos la matriz XM que contiene el vector Y, un vector de unos, esto con la función ones, indicando que el vector será de n por 1, donde n es el tamaño de Y, luego para iterar en las filas para obtener el vector p usamos la función size que nos dá el número de filas y columnas, posteriormente un ciclo for que va desde la fila 1 hasta numRows. De aqui el vector p se toma usando X(i,:) donde el simbolo ":" sirve para indicar todos los elementos, entonces diriamos que toma todos las columnas de la fila i, o lo que es lo mismo la fila i completa. Luego con la función transpose obtenemos la transpuesta de p y para obtener m basta con hacer multiplicación de vectores. Y para obtener la matriz S basta con sumar Mn de forma acumulativa.

```
1      %% Introducimos los valores X's y los valores de Y
2 -    tic
3 -    X =[ 1;
4           2;
5           3;
6           4];
7 -    Y = [6 ;
8           5;
9           7;
10          10];
11 -    b0 = ones(length(Y),1);
12 -    XM =[-1*Y b0 X];
13
14      %% Obetenemos la matriz M
15 -    [numRows,numCols] = size(XM);
16 -    S = 0;
17 -    for i = 1: numRows
18 -        p = XM(i,:);%cada vector p es una fila de XM
19 -        pt = transpose(p);
20 -        Mn = pt*p;%Matriz p al cuadrado
21 -        S = S + Mn; %Suma de matrices M, resulta en S
22 -    end
23
```

Figure 1: Obtención de S y Planteamiento del problema

4.2 Derivadas parciales de un polinomio

Para obtener la matriz de derivadas parciales hay que multiplocar cada fila po 2 como se muestra en la linea 26 del código, esto desde la fila 2 hasta la última, ya que no tomamos en cuenta la fila 1 como se explicó antes, luego eso lo agregamos a la matriz dS.

```
%% Se obtienen las derivadas parciales a partir de M
dS = [];
for i = 2: numCols
    element = 2*S(i,:);%Cada derivada parcial es una fila de la matriz exceptuando la primera fila, esto multiplicado por dos
    dS= [dS ;element]; % Vector con una expresion de derivada parcial
end
```

Figure 2: Obtención de dS

4.3 Resolución de sistema de ecuaciones

Primero separamos la matriz dS en la matriz B y A, donde la matriz B es la primera columna multiplicada por -1 y el resto de la matriz dS corresponde a la matriz A que contiene los coeficientes del sistema de ecuaciones, finalmente se saca la inversa de A y se multiplica por B para obtener la matriz C que contiene los valores de β_j

```
29    %% Separamos la matriz de derivadas parciales en los terminos independientes y el resto
30    [numRows,numCols] = size(dS);% se obtiene el numero de filas y de columnas
31    B = dS(:,1)*-1;% Vector de términos independientes, se multiplica por -1 ya que se esta despejando
32    A = dS(:,2:numCols) %Matriz de coeficientes
33    C = inversa(A)*B% Resolución de sistema de ecuaciones
34    Ynew= X*C(2:length(C)) + C(1)
35    desviacion = (Ynew-Y);
36    varianza = sum(desviacion.^2)/length(Y)
37    toc
```

Figure 3: Obtención de A, B y C

4.4 Matriz inversa

Para obtener la inversa se tiene la siguiente función, la cual se basa en el método de la Adjunta

Para obtener la inversa se tiene la siguiente función, la cual se basa en el método de la Adjunta el cual dice lo siguiente:

$$A^{-1} = \frac{Adj(A^T)}{det(A)} \tag{31}$$

```
function Inv = inversa(A)
%% La inversa se define como I = Adjunta(A transpuesta)/determinante de A
determinante = det(A); %obtencion del determinante de A
At= transpose(A); %obtención de la transpuesta de A
%% Obtenemos la adjunta de A transpuesta
[filas,columnas] = size(A);
Aadj = ones(filas); %generamos una matriz llena de 1's de nxn, es una matriz auxiliar
for i=1:filas %%Llenamos la matriz Adjunta en cada uno de sus elementos
    for j = 1: columnas

        if(mod(i,2)==0) %el factor depende de las posiciones de la matriz, ya sea 1 o -1 dependiendo de donde este
            if(mod(j,2)==1)
                factor = -1;% si la fila es par y columna es impar, el factor es -1
            else
                factor = 1;
            end
        else
            if(mod(j,2)==0)
                factor = -1; % si la fila es impar y la columna par, el factor es -1
            else
                factor=1;
            end
        end
        Cof = At;   % A partir de la matriz
        Cof(i,:) = []; %eliminamos la fila i
        Cof(:,j) = []; %eliminamos la columna j para obtener el cofactor
        Aadj(i,j)= factor*det(Cof); %el elemento i,j de la matriz adjunta es el determinante del cofactor multiplicador por el factor
    end
end
Inv = Aadj/determinante;%I = Adjunta(A transpuesta)/determinante de A
```

Figure 4: Función para obtener la inversa

5 Experimentos

5.1 Experimentos de baja dificultad

5.1.1 Experimento de 1 variable

X1	Y
1	6
2	5
3	7
4	10

```
A =

  Column 1

     8
    20

  Column 2

    20
    60


C =

    3.5000
    1.4000


Ynew =

    4.9000
    6.3000
    7.7000
    9.1000


varianza =

    1.0500

Elapsed time is 0.043732 seconds.
>>
```

Figure 5: Ejecución del experimento 1.1

5.1.2 Experimento de 1 variable

X1	Y
4	2
5	13
7	15

```
>> Proyecto3IA

A =

     6    32
    32   180


C =

  -10.5714
    3.8571


Ynew =

    4.8571
    8.7143
   16.4286


varianza =

    9.5238

Elapsed time is 0.002925 seconds.
>> 
```

Figure 6: Ejecución del experimento 1.2

5.1.3 Experimento de 2 variables

X1	X2	Y
3	2	1
4	14	12
5	15	24
6	9	11
12	6	14

```
C =

   -5.6537
    1.0006
    1.3098


Ynew =

   -0.0324
   16.6861
   18.9964
   12.1380
   14.2119


varianza =

    9.8801

Elapsed time is 0.004706 seconds.
>> |
```

Figure 7: Ejecución del experimento 1.3

5.2 Experimentos de media dificultad

5.2.1 Experimento de 4 variables

X1	X2	X3	X4	Y
2	11	8	9	11
4	2	6	6	21
5	4	9	8	12
6	14	7	3	6

```
C =


     0
     0
   -16
     0
     0



Ynew =


  -176
   -32
   -64
  -224



varianza =


   2.4114e+04


Elapsed time is 0.004210 seconds.
```

Figure 8: Ejecución del experimento 2.1

5.2.2 Experimento de 4 variables

X1	X2	X3	X4	Y
21	9	8	21	11
4	8	6	3	2
2	7	4	4	5
13	12	12	15	7

```
C =

  376.0000
   35.1875
  -72.5000
   43.2500
  -27.4688



Ynew =

  231.5938
  113.8438
    2.0000
   70.4062



varianza =

   1.6300e+04

Elapsed time is 0.046358 seconds.
>>
```

Figure 9: Ejecución del experimento 2.2

5.2.3 Experimento de 5 variables

X1	X2	X3	X4	X5	Y
2	14	12	9	6	11
6	15	6	7	3	2
9	11	8	8	5	6

```
C =

   1.0e+17 *

    1.5023
    0.2769
    0.2691
    0.4744
   -0.0897
   -0.9357


Ynew =

   1.0e+17 *

    5.0951
    6.6119
    5.3539


varianza =

   3.2780e+35

Elapsed time is 0.279214 seconds.
>>
```

Figure 10: Ejecución del experimento 2.3

5.2.4 Experimentos de alta dificultad

5.2.5 Experimento de 5 variables

X1	X2	X3	X4	X5	Y
4	7	11	24	7	16
6	8	24	14	8	5
13	6	15	15	9	14

```
C =

   1.0e+19 *

    4.5208
   -0.2652
   -1.4812
    0.0890
   -0.0015
    0.2707


Ynew =

   1.0e+19 *

   -4.0696
   -4.6391
   -4.0563


varianza =

   1.8179e+39

Elapsed time is 0.004347 seconds.
>>
```

Figure 11: Ejecución del experimento 3.1

5.2.6 Experimento de 5 variables

X1	X2	X3	X4	X5	Y
6	15	17	26	2	6
8	11	2	4	8	5
3	2	5	7	4	4

```
    5.2969
   -2.0036
   -0.3945
   -2.2849
   -0.5986
   -0.9044


Ynew =

   1.0e+17 *

   -6.8856
   -2.9270
   -2.0735


varianza =

   2.0093e+35

Elapsed time is 0.370963 seconds.
```

Figure 12: Ejecución del experimento 3.2

5.2.7 Experimento de 6 variables

X1	X2	X3	X4	X5	X6	Y
1	4	5	11	5	12	17
2	3	6	7	3	15	16
3	4	5	12	16	17	19

```
C =

   1.0e+19 *

   -5.7176
    0.3500
    1.0130
   -0.3502
   -0.2435
   -0.0307
   -0.2000


Ynew =

   1.0e+19 *

   -8.2991
   -8.8771
   -9.1801


varianza =

   7.7317e+39

Elapsed time is 0.046944 seconds.
```

Figure 13: Ejecución del experimento 3.3

6 Conclusiones

Se concluye que se logró implementar de manera exitosa el algoritmo de regresión lineal, el cual requirió ciertas herramientas matemáticas (de las cuales ya se tenía noción previa), mas sin embargo, fue de suma importancia recordar como realizarlas manualmente para poder plantear de manera correcta el código. Tales herramientas fueron el polinomio resultante de la diferencia de cuadradas, el cálculo de las derivadas parciales de un polinomio, así como saber resolver sistemas de ecuaciones; con ellas se le dió una base sólida al código que le permite calcular de manera exitosa la regresión lineal.

7 Referencias

- "¿Qué es la regresión lineal?" MATLAB SIMULINK. Recuperado el 23 de Noviembre de: https://la.mathworks.com/discover
  regression.html

Código

```
function Inv = inversa(A)
%% La inversa se define como I = Adjunta(A transpuesta)/determinante de A
determinante = det(A); %obtencion del determinante de A
At= transpose(A); %obtenci n de la transpuesta de A
%% Obtenemos la adjunta de A transpuesta
[filas,columnas] = size(A);
Aadj = ones(filas); %generamos una matriz llena de 1's de nxn, es una matriz auxiliar
for i=1:filas %%Llenamos la matriz Adjunta en cada uno de sus elementos
    for j = 1: columnas

        if(mod(i,2)==0) %el factor depende de las posiciones de la matriz, ya sea 1 o -1 dependiendo de donde este
            if(mod(j,2)==1)
                factor = -1;% si la fila es par y columna es impar, el factor es -1
            else
                factor = 1;
            end
        else
            if(mod(j,2)==0)
                factor = -1; % si la fila es impar y la columna par, el factor es -1
            else
                factor=1;
            end
        end
        Cof = At;   % A partir de la matriz
        Cof(i,:) = []; %eliminamos la fila i
        Cof(:,j) = []; %eliminamos la columna j para obtener el cofactor
        Aadj(i,j)= factor*det(Cof); %el elemento i,j de la matriz adjunta es el determinante del cofactor multiplicador por el factor
    end
end
Inv = Aadj/determinante;%I = Adjunta(A transpuesta)/determinante de A
Comprobacion = Inv*A%%Si al multiplicar la inversa sale la identidad se cumple la inversa
end
```

```
1  %% Introducimos los valores X's y los valores de Y
2  tic
3  X =[ 1 4 5;
4       7 6 7;
5       4 7 9;
6       4 9 1];
7  Y = [2 ;
8       4;
9       6;
10      8];
11  b0 = ones(length(Y),1);
12  XM =[-1*Y b0 X];
13
14  %% Obetenemos la matriz M
15 [numRows,numCols] = size(XM);
16 S = 0;
17 for i = 1: numRows
18     p = XM(i,:);%cada vector p es una fila de XM
19     pt = transpose(p);
20     Mn = pt*p;%Matriz p al cuadrado
21     S = S + Mn; %Suma de matrices M, resulta en S
22 end
23
24  %% Se obtienen las derivadas parciales a partir de M
25  dS = [];
26  for i = 2: numCols
27    element = 2*S(i,:);%Cada derivada parcial es una fila de la matriz exceptuando
         la primera fila, esto multiplicado por dos
28    dS= [dS ;element]; % Vector con una expresion de derivada parcial
29 end
30
31 %% Separamos la matriz de derivadas parciales en los terminos independientes y el
      resto
32 [numRows,numCols] = size(dS);% se obtiene el numero de filas y de columnas
33 B = dS(:,1)*-1;% Vector de t rminos independientes, se multiplica por -1 ya que se
       esta despejando
34 A = dS(:,2:numCols) %Matriz de coeficientes
35 C = inv(A)*B% Resoluci n de sistema de ecuaciones
36 Ynew= X*C(2:length(C)) + C(1)
37 desviacion = (Ynew-Y);
38 varianza = sum(desviacion.^2)/length(Y)
39 toc
```

De la línea 36 a 38 Nos sirve para calcular la desviación.

Inteligencia Artificial

Proyecto 3

Saldaña Hernandez Norman Brandon
Guerrero Ferrusca Francisco
González Lorenzo Daniela Michel
Tenorio Veloz Alisson Dafne

November 24, 2021

Planteamiento del problema

Implementar un algoritmo de regresión lineal generalizada con las siguientes características:

- Implementar una función para calcular el polinomio resultante de la diferencia de cuadrados.
- Implementar una función para calcular la derivada parcial de un polinomio al cuadrado-
- Implementa un algoritmo que encuentre la solución de un sistema de ecuaciones.

Solución del problema

La solución se llevó a cabo mediante la programación de las herramientas anteriormente mencionadas en Matlab, ya que es un software que permite programación en alto nivel orientado a manejo de datos, sobre todo de matrices, en este caso la sintaxis resulta sencilla y el manejo de arreglos de datos no requiere un manejo de memoria implicito. El software está optimizado cuando se usan solo matrices.

Definición de regresión lineal generalizada

$$\mathbf{S}(\beta_j) : \sum_{i=1}^{n}(y_i - (\beta_0 + \sum_{j=1}^{m} \beta_j x_{i,j}))^2$$

$$\frac{dS}{d\beta_j} = 0$$

$$y = \beta_0 + \sum_{j=1}^{m} \beta_j X_j$$

Entradas del problema

$$\begin{bmatrix} \mathbf{y} \\ 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \begin{bmatrix} \mathbf{x_1} \\ 6 \\ 5 \\ 7 \\ 10 \end{bmatrix} \begin{bmatrix} \mathbf{x_2} \\ 16 \\ 5 \\ 31 \\ 9 \end{bmatrix} \cdots \begin{bmatrix} \mathbf{x_{m-1}} \\ 2 \\ 5 \\ 1 \\ 29 \end{bmatrix} \begin{bmatrix} \mathbf{x_m} \\ 15 \\ 7 \\ 3 \\ 1 \end{bmatrix}$$

Vector P

$$(y_i - (\beta_0 + \sum_{j=1}^{m} \beta_j x_{i,j}))$$

$$p = \begin{bmatrix} -\mathbf{y} & \beta_0 & \beta_1 & \beta_2 & \cdots & \beta_{j-1} & \beta_j \\ y_1 & 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,j-1} & x_{1,j} \end{bmatrix} = \begin{bmatrix} -\mathbf{y} & \beta_0 & \beta_1 & \beta_2 & \cdots & \beta_{j-1} & \beta_j \\ 1 & 1 & 6 & 16 & \cdots & 2 & 15 \end{bmatrix}$$

Matriz XM, arreglo de vectores P

$$\sum_{i=1}^{n}(y_i - (\beta_0 + \sum_{j=1}^{m}\beta_j x_{i,j}))$$

$$\begin{pmatrix} -y_{1,1} & 1 & x_{1,1} & \cdots & x_{n-1,m-1} & x_{n,m} \\ -y_{2,1} & 1 & x_{2,1} & \cdots & x_{n-1,m-1} & x_{n,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -y_{n-1,1} & 1 & x_{n-1,1} & \cdots & x_{n-1,m-1} & x_{n,m} \\ -y_{n,1} & 1 & x_{n,1} & \cdots & x_{n-1,m-1} & x_{n,m} \end{pmatrix}$$

Ejemplo de matriz XM

$$\begin{pmatrix} y & x \\ 1 & 6 \\ 2 & 5 \\ 3 & 7 \\ 4 & 10 \end{pmatrix} \rightarrow \begin{pmatrix} y & \beta_0 & \beta_1 \\ -1 & 1 & 6 \\ -2 & 1 & 5 \\ -3 & 1 & 7 \\ -4 & 1 & 10 \end{pmatrix}$$

Vector p al cuadrado = Matriz M

$$p = \begin{bmatrix} a_1 & a_2 & \cdots & a_{m-1} & a_m \end{bmatrix}$$

$$p^2 = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{pmatrix} a_1a_1 & a_1a_2 & a_1a_3 \\ a_2a_1 & a_2a_2 & a_2a_3 \\ a_3a_1 & a_3a_2 & a_3a_3 \end{pmatrix} = M$$

$$p_\beta^2 = \begin{bmatrix} y & \beta_0 & \beta_1 \end{bmatrix} \begin{bmatrix} y \\ \beta_0 \\ \beta_1 \end{bmatrix} = \begin{pmatrix} yy & y\beta_0 & y\beta_1 \\ \beta_0 y & \beta_0\beta_0 & \beta_0\beta_1 \\ \beta_1 y & \beta_1\beta_0 & \beta_1\beta_1 \end{pmatrix}$$

$$Mi = p_i^2 = p_i * p_i^T$$

Matriz S como suma de matrices M, respresentación de polimonios al cuadrado

$$\begin{pmatrix} y & x \\ 1 & 6 \\ 2 & 5 \\ 3 & 7 \\ 4 & 10 \end{pmatrix} \rightarrow p_1 = \begin{bmatrix} 1 & 1 & 6 \end{bmatrix} \rightarrow p^2 = \begin{bmatrix} -1 & 1 & 6 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 6 \end{bmatrix} = \begin{pmatrix} (1)(1) & (1)(1) & (1)(6) \\ (1)(1) & (1)(1) & (1)(6) \\ (6)(1) & (6)(1) & (6)(6) \end{pmatrix} = \begin{pmatrix} 1 & -1 & -6 \\ -1 & 1 & 6 \\ -6 & 6 & 36 \end{pmatrix}$$

$$S_1 = (y_1 - (\beta_0 + \beta_1 x_{1,1}))^2 = (1 - (\beta_0 + 6\beta_1))^2 = (-1 + \beta_0 + \beta_1)^2 =$$

$$= 1 - \beta_0 - 6\beta_1 - \beta_0 + \beta_0^2 + 6\beta_0\beta_1 - 6\beta_1 + 6\beta 1\beta_0 + 36\beta_1^2$$
$$= 1 - 2\beta_0 - 12\beta_1 + \beta_0^2 + 12\beta_0\beta_1 + 36\beta_1^2$$

Del espacio de matrices al espacio de polinomios

$$M(1,1) + (M(1,2) + M(2,1))\beta_0 + (M(1,3) + M(3,1))\beta_1 + (M(2,3) + M(3,2))\beta_0\beta_1 + M(2,2)\beta_0^2 + M(3,3)\beta_1^2$$

$$1 + (-1-1)\beta_0 + (-6-6)\beta_1 + (6+6)\beta_0\beta_1 + 1\beta_0^2 + 36\beta_1^2$$

$$1 - 2\beta_0 - 12\beta_1 + 12\beta_0\beta_1 + 1\beta_0^2 + 36\beta_1^2$$

Definición de S

$$
\begin{aligned}
S = & \begin{bmatrix} y_{1,1} & 1 & x_{1,1} & \cdots & x_{n-1,m-1} & x_{n,m} \end{bmatrix}^2 + \begin{bmatrix} y_{2,1} & 1 & x_{2,1} & \cdots & x_{n-1,m-1} & x_{n,m}
\\
& + \begin{bmatrix} y_{n-1,1} & 1 & x_{n-1,1} & \cdots & x_{n-1,m-1} & x_{n,m} \end{bmatrix} + \begin{bmatrix} y_{n,1} & 1 & x_{n,1} & \cdots & x_{n,m-1} & x_{n}
\end{aligned}
$$

$$S = M_1 + M_2 + M_3 + \cdots + M_{i-1} + M_i$$

Derivadas parciales

$$\begin{pmatrix} yy & y\beta_0 & y\beta_1 \\ \beta_0 y & \beta_0\beta_0 & \beta_0\beta_1 \\ \beta_1 y & \beta_1\beta_0 & \beta_1\beta_1 \end{pmatrix}$$

$$S = M_1 + M_2 + M_3 + M_4 = \begin{bmatrix} 30 & -10 & -77 \\ -10 & 4 & 28 \\ -77 & 28 & 210 \end{bmatrix}$$

Entonces la derivada parcial de **S** con respecto a β_0 sería 2*[-10 4 28] =[-20 8 28], donde asignaremos a cada elemento de este vector obtenido su respectiva variable y nos quedaría: $-20 + 8\beta_0 + 28\beta 1$
Para la derivada de **S** con respecto a β_1 quedaría: $-144 + 56\beta_0 + 420\beta_1$ y a partir de esto podemos construir una matriz **dS** que contenga todas las derivadas parciales, solo bastaría con multiplicar por 2 la matriz S y quitar la primera fila.
Para este caso **dS** quedaría así:

$$dS = \begin{bmatrix} -20 & 8 & 28 \\ -144 & 56 & 420 \end{bmatrix}$$

Resolución al sistema de ecuaciones

$$dS \xrightarrow[descomponiendo]{} B = \begin{bmatrix} -20 \\ -144 \end{bmatrix} A = \begin{bmatrix} 8\beta_0 & 28\beta_1 \\ 56\beta_0 & 420\beta_1 \end{bmatrix}$$

$$C = inv(A) * B$$

$$A^{-1} = \frac{Adj(A^T)}{det(A)}$$

Desarrollo del código (Herramientas matemáticas)

Polinomio al cuadrado

```
%% Introducimos los valores X's y los valores de Y
tic
X =[ 1;
     2;
     3;
     4];
Y = [6 ;
     5;
     7;
     10];
 b0 = ones(length(Y),1);
 XM =[-1*Y b0 X];

 %% Obetenemos la matriz M
[numRows,numCols] = size(XM);
S = 0;
for i = 1: numRows
    p = XM(i,:);%cada vector p es una fila de XM
    pt = transpose(p);
    Mn = pt*p;%Matriz p al cuadrado
    S = S + Mn; %Suma de matrices M, resulta en S
end

```

Figure: Obtención de S y Planteamiento del problema

Desarrollo del código (Herramientas matemáticas)

Derivadas parciales de un polinomio.

```
%% Se obtienen las derivadas parciales a partir de M
dS = [];
for i = 2: numCols
  element = 2*S(i,:);%Cada derivada parcial es una fila de la matriz exceptuando la prim
  dS= [dS ;element]; % Vector con una expresion de derivada parcial
end
```

Figure: Obtención de dS

Desarrollo del código (Herramientas matemáticas)

Resolución del sistema de ecuaciones.

```
%% Separamos la matriz de derivadas parciales en los terminos independientes y el resto
[numRows,numCols] = size(dS);% se obtiene el numero de filas y de columnas
B = dS(:,1)*-1;% Vector de términos independientes, se multiplica por -1 ya que se esta despejando
A = dS(:,2:numCols); %Matriz de coeficientes
C = inversa(A)*B% Resolución de sistema de ecuaciones
toc
```

Figure: Obtención de A, B y C

Desarrollo del código (Herramientas matemáticas)

Desviación del algoritmo.

```
Ynew= X*C(2:length(C)) + C(1)
desviacion = (Ynew-Y);
varianza = sum(desviacion.^2)/length(Y)
```

Figure: Obtención de la desviación

Desarrollo del código (Herramientas matemáticas)

Matriz inversa.

```
function Inv = inversa(A)
%% La inversa se define como I = Adjunta(A transpuesta)/determinante de A
determinante = det(A); %obtencion del determinante de A
At= transpose(A); %obtención de la transpuesta de A
%% Obtenemos la adjunta de A transpuesta
[filas,columnas] = size(A);
Aadj = ones(filas); %generamos una matriz llena de 1's de nxn, es una matriz auxiliar
for i=1:filas %%Llenamos la matriz Adjunta en cada uno de sus elementos
    for j = 1: columnas

        if(mod(i,2)==0) %el factor depende de las posiciones de la matriz, ya sea 1 o -1 dependiendo de donde este
            if(mod(j,2)==1)
                factor = -1;% si la fila es par y columna es impar, el factor es -1
            else
                factor = 1;
            end
        else
            if(mod(j,2)==0)
                factor = -1; % si la fila es impar y la columna par, el factor es -1
            else
                factor=1;
            end
        end
        Cof = At;   % A partir de la matriz
        Cof(i,:) = []; %eliminamos la fila i
        Cof(:,j) = []; %eliminamos la columna j para obtener el cofactor
        Aadj(i,j)= factor*det(Cof); %el elemento i,j de la matriz adjunta es el determinante del cofactor multiplicador por el facto
    end
end
Inv = Aadj/determinante;%I = Adjunta(A transpuesta)/determinante de A
```

Figure: Función para obtener la inversa

Experimentos de baja dificultad

Experimento de una variable

X1	Y
1	6
2	5
3	7
4	10

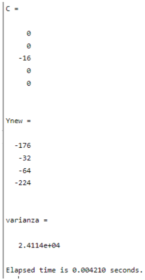
```
A =

  Column 1

     8
    20

  Column 2

    20
    60


C =

    3.5000
    1.4000


Ynew =

    4.9000
    6.3000
    7.7000
    9.1000


varianza =

    1.0500

Elapsed time is 0.043732 seconds.
>>
```

Figure: Ejecución del experimento 1.1

Experimentos de media dificultad

Experimento de 4 variables

X1	X2	X3	X4	Y
2	11	8	9	11
4	2	6	6	21
5	4	9	8	12
6	14	7	3	6



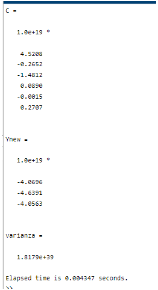
```
C =

     0
     0
   -16
     0
     0


Ynew =

  -176
   -32
   -64
  -224


varianza =

   2.4114e+04

Elapsed time is 0.004210 seconds.
```

Figure: Ejecución del experimento 2.1

Experimentos de alta dificultad

Experimento de 5 variables

X1	X2	X3	X4	X5	Y
4	7	11	24	7	16
6	8	24	14	8	5
13	6	15	15	9	14



```
C =

   1.0e+19 *

    4.5208
   -0.2652
   -1.4812
    0.0890
   -0.0015
    0.2707


Ynew =

   1.0e+19 *

   -4.0696
   -4.6391
   -4.0563


varianza =

   1.8179e+39

Elapsed time is 0.004347 seconds.
>>
```

Figure: Ejecución del experimento 3.1

Conclusiones

Se concluye que se logró implementar de manera exitosa el algoritmo de regresión lineal, el cual requirió ciertas herramientas matemáticas (de las cuales ya se tenía noción previa), mas sin embargo, fue de suma importancia recordar como realizarlas manualmente para poder plantear de manera correcta el código. Tales herramientas fueron el polinomio resultante de la diferencia de cuadradas, el cálculo de las derivadas parciales de un polinomio, así como saber resolver sistemas de ecuaciones; con ellas se le dió una base sólida al código que le permite calcular de manera exitosa la regresión lineal.

Referencias

- "¿Qué es la regresión lineal?" MATLAB SIMULINK. Recuperado el 23 de Noviembre de: https://la.mathworks.com/discovery/linear-regression.html