PRÁCTICA 4: MACHINE LEARNING - REGRESIÓN

Estudiante 1: Ana Isabel Grima Montesa
Estudiante 2: Eloy Medrano Gil
Grupo: 2

1 CUESTIONES.

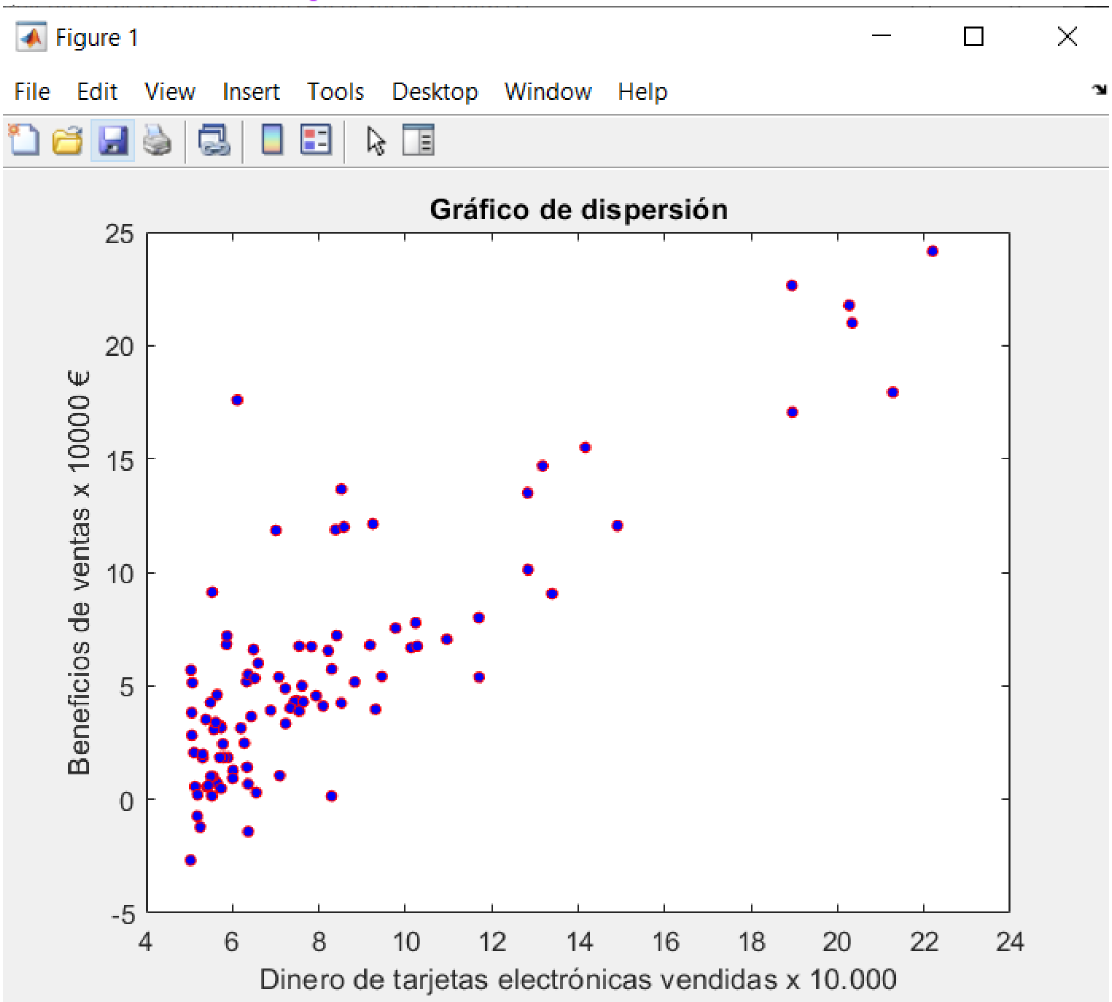
1.1 CUESTIÓN 1.

1.1.1 INTRODUCCIÓN.

Esta cuestión se basa en la resolución de un ejercicio, en el que implementará una regresión lineal con una variable, empleándose Matlab, para predecir los beneficios por ventas de una empresa.
Supóngase que somos es el CEO de una empresa que fábrica tarjetas electrónicas para el control de motores de c.c. y está considerando en modificar, una vez más su producción debido a las demandas fluctuantes del mercado. La empresa dispone de históricos y bases de datos que relacionan los beneficios obtenidos en función del número de unidades de tarjetas vendidas.
Se deberán utilizar estos datos para ayudar al CEO a determinar el beneficio que se obtendrá, a partir del número estimado de ventas de tarjetas, para que, de esta forma, el departamento financiero de la empresa desarrolle el plan futuro de viabilidad de la empresa.
Para ello se proporciona un archivo denominado cuestion41_data.txt que contiene el conjunto de datos de entrenamiento para dicho problema de regresión lineal, como ya se ha comentado, se trata de la base de datos que relacionan los beneficios obtenidos, en tiempos pasados, en función del número de ventas del producto. La primera columna es el número de tarjetas vendidas y la segunda columna es el beneficio obtenido debido a esas ventas. Un valor negativo de dicho beneficio indicaría la entrada en pérdidas para la empresa

1.1.2 GRÁFICO DE DISPERSIÓN PARA LOS DATOS.

```
datos=load('cuestion41_data.txt');
x=datos(:,1);
y=datos(:,2);
plot(x,y,'ro','markersize',4,'markerfacecolor','b');

title('Gráfico de dispersión')
ylabel('Beneficios de ventas x 10000 €');
xlabel('Dinero de tarjetas electrónicas vendidas x 10.000');
```

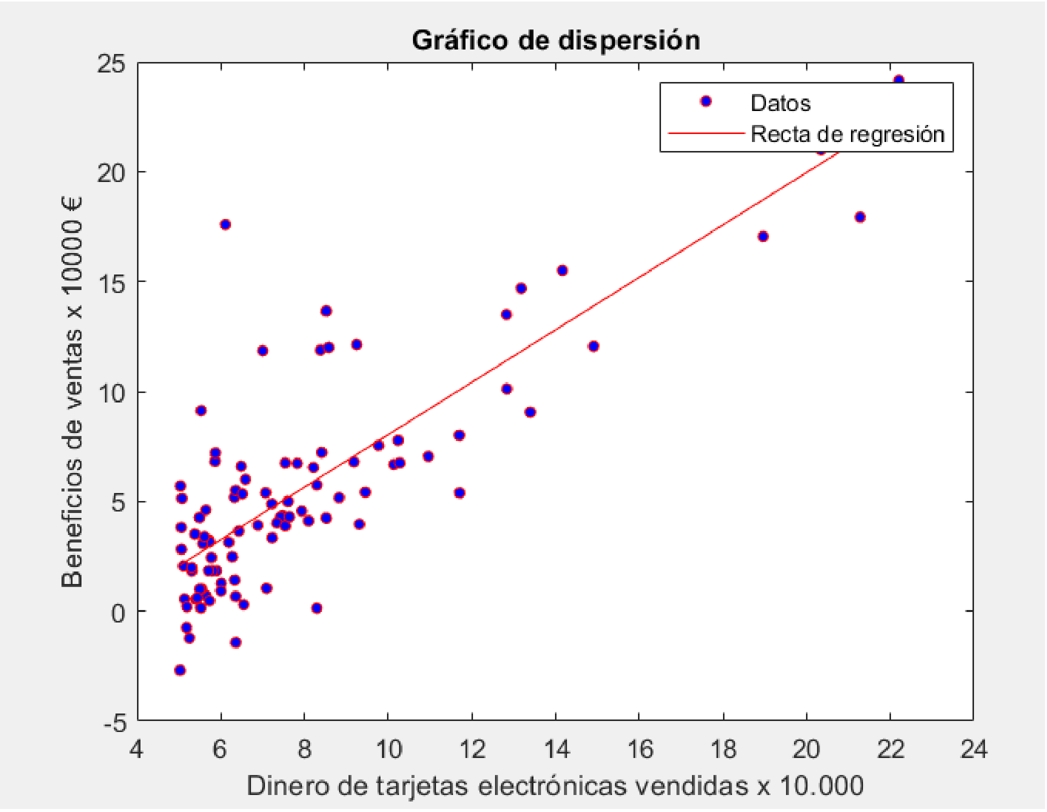


Mediante el código escrito anteriormente se obtiene la gráfica de dispersión de datos. En esta gráfica se puede observar que en la mayoría de los casos la empresa ha obtenido beneficios positivos.

1.1.3 OBTENCIÓN DE LA RECTA DE REGRESIÓN LINEAL.

Recta de regresión con polyfit

```
>> p=polyfit(x,y,1)

p =

    1.1930   -3.8958

>> z=polyval(p,x); % z = theta0 + theta1*x
>> hold on
>> plot(x,z,'r')
>> legend('Datos','Recta de regresión')
>> hold off
```



theta0 = -3.8958

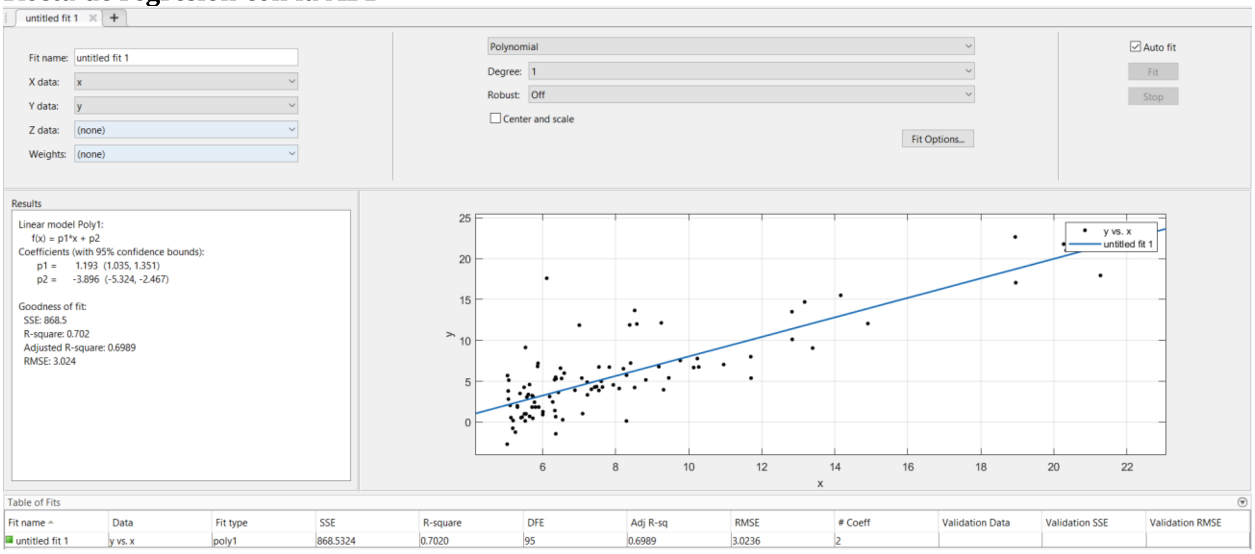
theta1= 1.1930

y= theta0+theta1*x

Recta de regresión con Fitlm

```
>> lm=fitlm(x,y)

lm =


Linear regression model:
    y ~ 1 + x1

Estimated Coefficients:
                   Estimate       SE        tStat       pValue  
                   ________    ________    _______    __________

    (Intercept)    -3.8958      0.71948    -5.4147    4.6079e-07
    x1               1.193     0.079744     14.961    1.0232e-26


Number of observations: 97, Error degrees of freedom: 95
Root Mean Squared Error: 3.02
R-squared: 0.702,  Adjusted R-Squared: 0.699
F-statistic vs. constant model: 224, p-value = 1.02e-26
```

Recta de regresión con la APP



Con esta app podemos observar que ambas expresiones de la recta nos han salido iguales y con un error cuadrático del 3,02%

1.1.4 PREDICCIÓN.

f(x) = p1*x + p2

theta1= p1

theta0 = p2

```
>> p=polyfit(x,y,1)

p =

    1.1930   -3.8958

>> s1=polyval(p,35000) % s1 = theta0 + theta1*35000

s1 =

   4.1752e+04

>> s2=polyval(p,70000) % s2 = theta0 + theta1*7000

s2 =

   8.3508e+04
```

Con este modelo de predicción observamos que con la venta de 35000 tarjetas electrónicas la empresa obtendría un beneficio de 41752€ y con la venta de 70000 tarjetas electrónicas obtendría un beneficio de 83508€

1.2 CUESTIÓN 2.

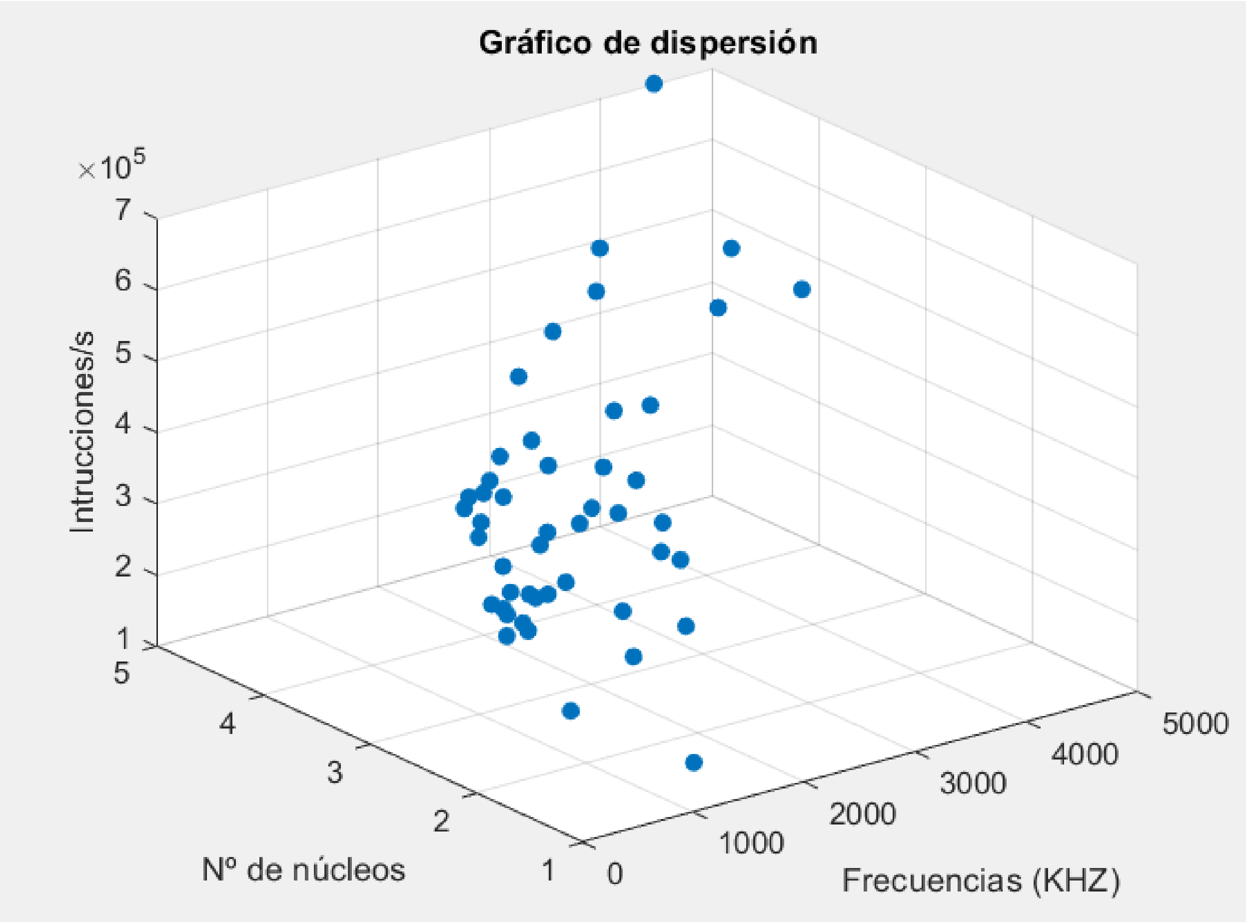
1.2.1 INTRODUCCIÓN.

Esta cuestión se basa en la resolución de un ejercicio, en el que se implementará una regresión lineal con múltiples variables, empleándose Matlab, para predecir las instrucciones por segundo que podrá ejecutar un microprocesador de bajo coste.

El equipo técnico de una compañía que fabrica microprocesadores de bajo coste necesita conocer las instrucciones que podrá ejecutar un microprocesador en un segundo antes de ser fabricado, para analizar si es necesario seguir con la misma tecnología de fabricación o cambiar a otra más novedosa. Una forma de hacerlo sería disponer de una base de datos con información recopilada de todos los modelos que ha fabricado hasta la fecha y hacer un modelo para predecir el número de instrucciones por segundo que puede ejecutar un microprocesador.

1.2.2 GRÁFICO DE DISPERSIÓN PARA LOS DATOS.

```
>> tbl=readtable('cuestion42_data.txt');
>> datos=table2array(tbl);
>> x1=datos(:,1);
>> x2=datos(:,2);
>> y=datos(:,3);
>> scatter3(x1,x2,y,'filled')
>> title('Gráfico de dispersión');
>> xlabel('Frecuencias (KHZ)');
>> ylabel('Nº de núcleos');
>> zlabel('Intrucciones/s');
```



En esta gráfica de dispersión de datos podemos observar que cuanto mayor es el número de núcleos, mayor es el número de instrucciones ejecutadas en un segundo, lo que significa que posee una mayor frecuencia de funcionamiento.

1.2.3 OBTENCIÓN DE LA RECTA DE REGRESIÓN LINEAL MULTIVARIABLE.

Regresión con fitlm

```
>> mdl=fitlm(tbl)


mdl =


Linear regression model:
    Instrucciones ~ 1 + Frecuencia + Nucleos

Estimated Coefficients:
                   Estimate      SE       tStat       pValue
                   ________    ______    ________    __________

    (Intercept)      89598      41767      2.1452      0.037499
    Frecuencia      139.21     14.795      9.4092    4.2223e-12
    Nucleos          -8738      15451    -0.56554       0.57458


Number of observations: 47, Error degrees of freedom: 44
Root Mean Squared Error: 6.61e+04
R-squared: 0.733,  Adjusted R-Squared: 0.721
F-statistic vs. constant model: 60.4, p-value = 2.43e-13
```

Regresión con la ecuación normal

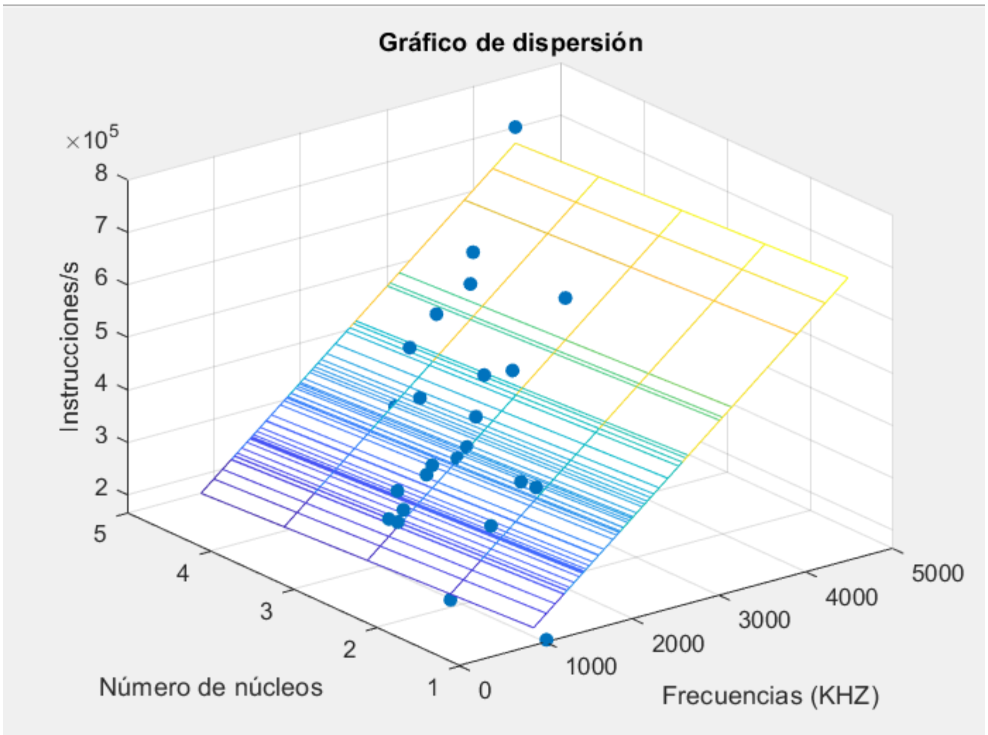
```
>> X=datos(:, 1:2);
>> n=length(y);
>> X1=[ones(n,1) X];
>> theta=pinv(X1' *X1)*X1' *y

theta =

   1.0e+04 *

    8.9598
    0.0139
   -0.8738
```

Regresión con regress

```
X=[ones(size(x1)) x1 x2];
b=regress(y,X);
```

Plano regresión lineal

```
scatter3(x1,x2,y,'filled')
title('Gráfico de dispersión');
xlabel('Frecuencias (KHZ)');
ylabel('Número de núcleos');
zlabel('Instrucciones/s');
hold on
[X1FIT,X2FIT]=meshgrid(x1,x2);
coeficientes=table2array(mdl.Coefficients);
YFIT=coeficientes(1,1)+coeficientes(2,1)*X1FIT+coeficientes(3,1)*X2FIT;
mesh(X1FIT,X2FIT,YFIT);
hold off
```



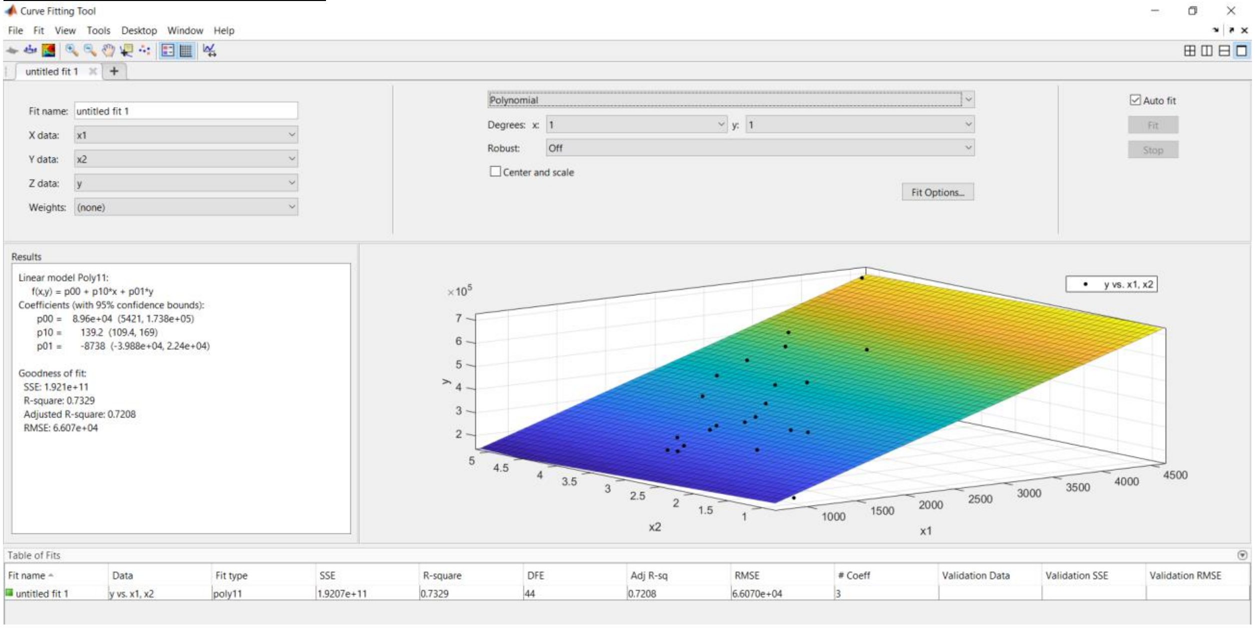
La ecuación nos quedaría z=theta0+theta1*x+theta2*y. La cual coincidiría para cada caso y se obtendría el plano de regresión lineal como el de la imagen.

1.2.4 PREDICCIÓN.

```
>> I=theta(1)+1650*theta(2)+3*theta(3)

I =

   2.9308e+05
```

Según el modelo que hemos generado, la predicción de instrucciones por segundo que realizará el microprocesador a 1650Hz y con 3 núcleos será de 293080 instrucciones por segundo.

1.2.5 APP DE MATLAB.



Según el código nos salen unos valores para theta0=8.9598e+04, theta1=0.0139e+04 y theta2=-0.8738e+04 y un erro cuadrático de 0.733, un error cuadrático ajustado de 0.721 y un RMSE=6.61e+04.Mientras que en la app nos salen valores de theta0=8.96e+04, theta1=0.01392e+04 y theta2=-0.8738e+04 y un erro cuadrático de 0.7329, un error cuadrático ajustado de 0.7208 y un RMSE=6.607e+04

1.3 CUESTIÓN 3.

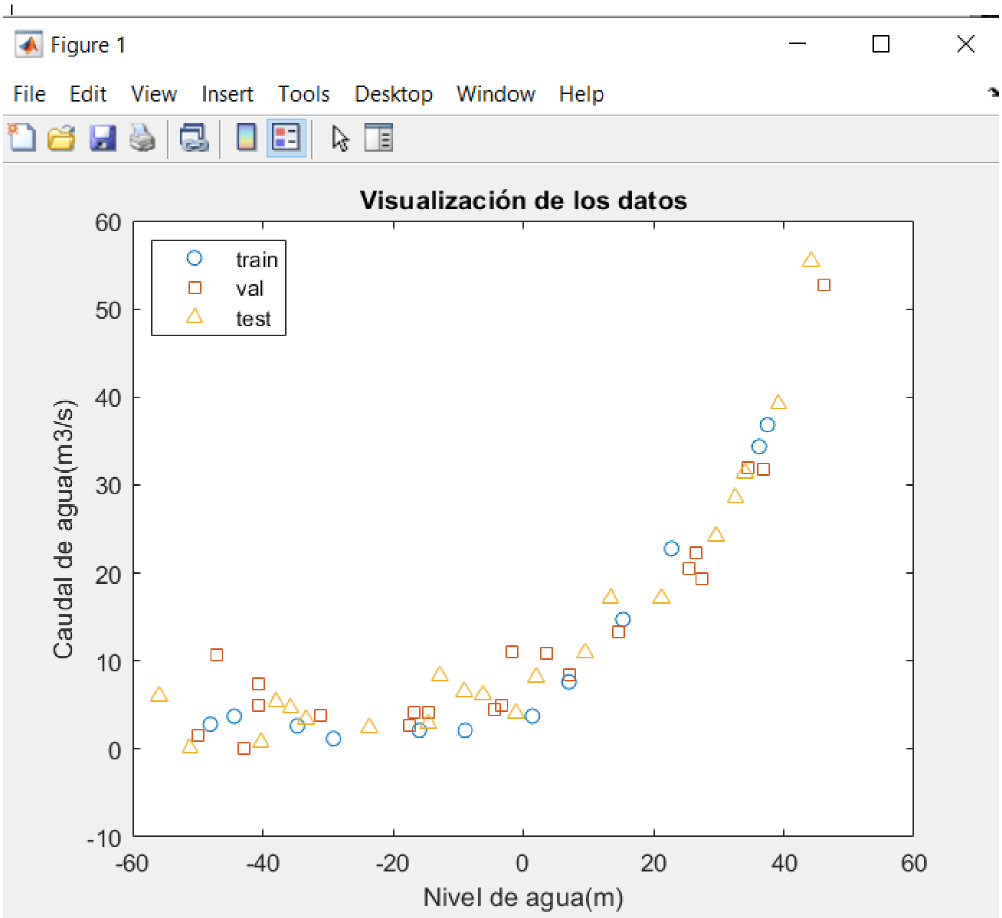
1.3.1 INTRODUCCIÓN.

Se implementará una regresión polinomial para predecir la cantidad de agua que sale por la presa de un embalse en función del cambio de nivel de agua que se produce en dicho embalse.

Al cargar el fichero cuestion43_data.mat en el entorno de trabajo de Matlab, se dispondrá de los conjuntos de datos siguientes, generados de forma aleatoria del total de datos disponibles:
— Un conjunto de entrenamiento como base para la obtención de varios modelos: “xtrain” e “ytrain”.
— Un conjunto de validación cruzada para evaluar los modelos anteriores y determinar el mejor, “xval”, e “yval”. — Un conjunto de prueba o test para evaluar el rendimiento del modelo obtenido. Estos son ejemplos "invisibles" que el modelo no vio durante el entrenamiento: “xtest” e “ytest”.

1.3.2 VISUALIZACIÓN DE LOS DATOS.

```
load("cuestion43_data.mat");
plot(xtrain,ytrain,"o",xval,yval,"s",xtest,ytest,'^');
title('Visualización de los datos')
legend("train",'val','test','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```



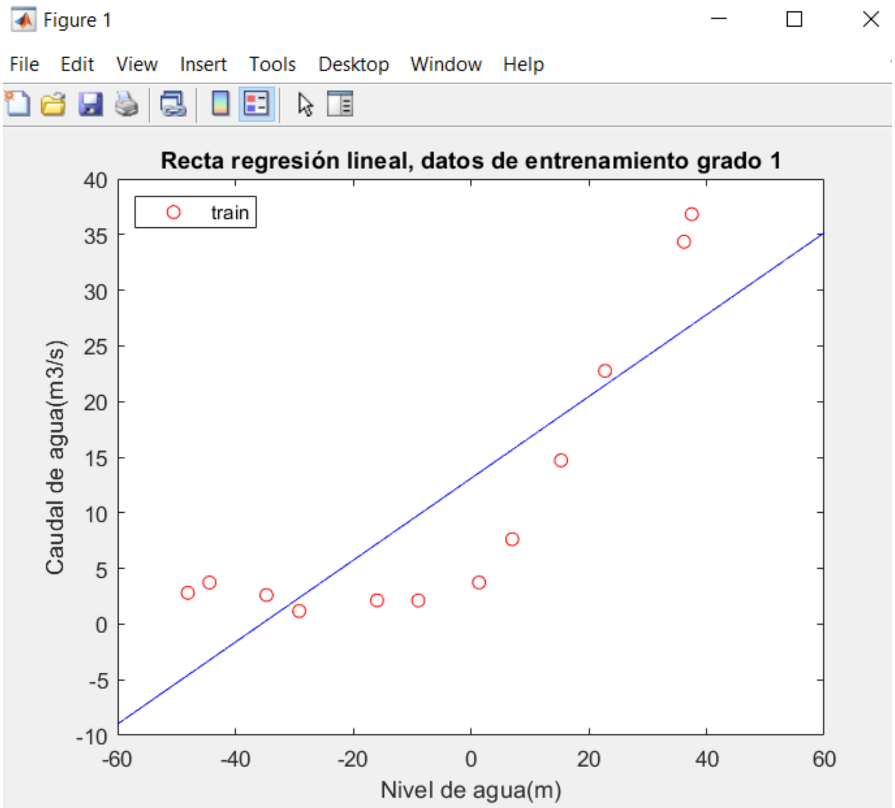
En la imagen antesior se puede observar la gráfica de dispersión de los 3 conjuntos de datos. Podemos distinguirlos ya que tienen una simbología diferente.

1.3.3 REGRESIÓN LINEAL.

Entrenamiento grado 1

```
>> polymodel1=polyfitn(xtrain,ytrain,1)

polymodel1 =

  struct with fields:

      ModelTerms: [2×1 double]
    Coefficients: [0.3678 13.0879]
    ParameterVar: [0.0054 4.6154]
    ParameterStd: [0.0737 2.1483]
             DoF: 10
               p: [5.4706e-04 1.1691e-04]
              R2: 0.7133
      AdjustedR2: 0.6846
            RMSE: 6.6894
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred1 = polyvaln(polymodel1,x);
plot(xtrain,ytrain,'ro',x,ypred1,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 1');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```



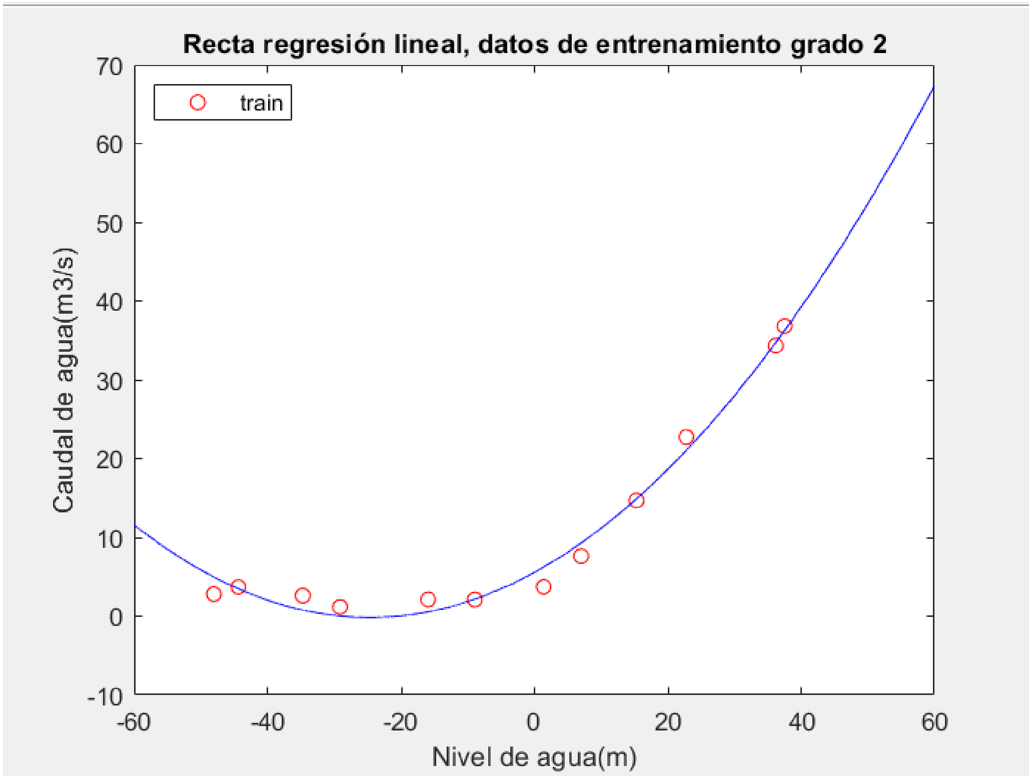
Con la regresión lineal y entrenamiento de grado 1 nos queda una ecuación como la siguiente: y=thetha0*x+theta1

1.3.4 MODELOS DE REGRESIÓN POLINOMIAL.

Entrenamiento grado 2

```
>> polymodel2=polyfitn(xtrain,ytrain,2)

polymodel2 =

  struct with fields:

      ModelTerms: [3×1 double]
    Coefficients: [0.0094 0.4652 5.5842]
    ParameterVar: [4.7062e-07 3.2538e-04 0.5319]
    ParameterStd: [6.8601e-04 0.0180 0.7293]
             DoF: 9
               p: [2.4173e-07 9.5431e-10 3.1367e-05]
              R2: 0.9869
      AdjustedR2: 0.9840
            RMSE: 1.4273
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred2 = polyvaln(polymodel2,x);
plot(xtrain,ytrain,'ro',x,ypred2,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 2');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

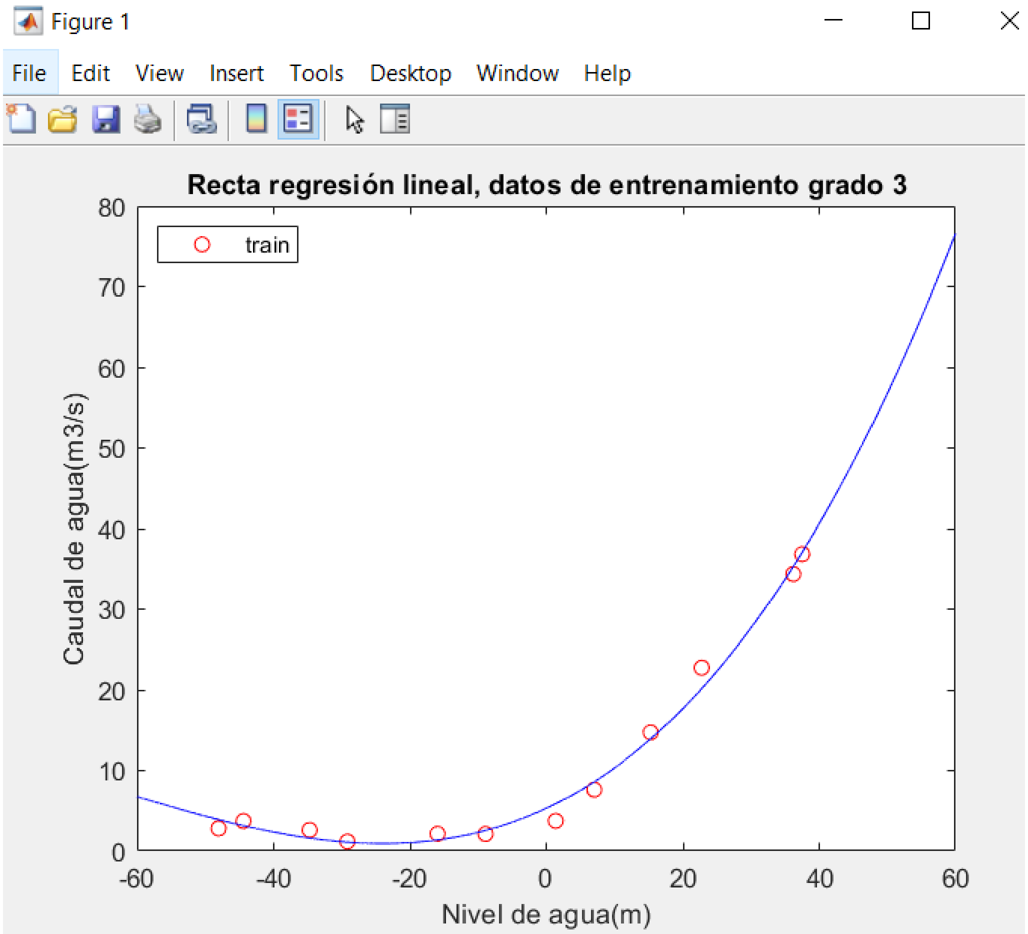


Con la regresión lineal y entrenamiento de grado 2 nos queda una ecuación como la siguiente: y=thetha0*x^2+theta1*x+theta2

Entrenamiento grado 3

```
>> polymodel3=polyfitn(xtrain,ytrain,3)

polymodel3 =

  struct with fields:

      ModelTerms: [4×1 double]
    Coefficients: [5.1814e-05 0.0101 0.3951 5.3185]
     ParameterVar: [7.9525e-10 5.0778e-07 0.0017 0.4417]
     ParameterStd: [2.8200e-05 7.1259e-04 0.0414 0.6646]
              DoF: 8
                p: [0.1035 5.9775e-07 1.1973e-05 4.3586e-05]
               R2: 0.9908
       AdjustedR2: 0.9874
             RMSE: 1.1970
         VarNames: {'X1'}
```

```
>> x=(-60:0.01:60);
>> ypred3 = polyvaln(polymodel3,x);
>> plot(xtrain,ytrain,'ro',x,ypred3,'b-')
>> title('Recta regresión lineal, datos de entrenamiento grado 3');
>> legend("train",'Location','northwest','Orientation','vertical');
>> xlabel('Nivel de agua(m)');
>> ylabel('Caudal de agua(m3/s)');
```

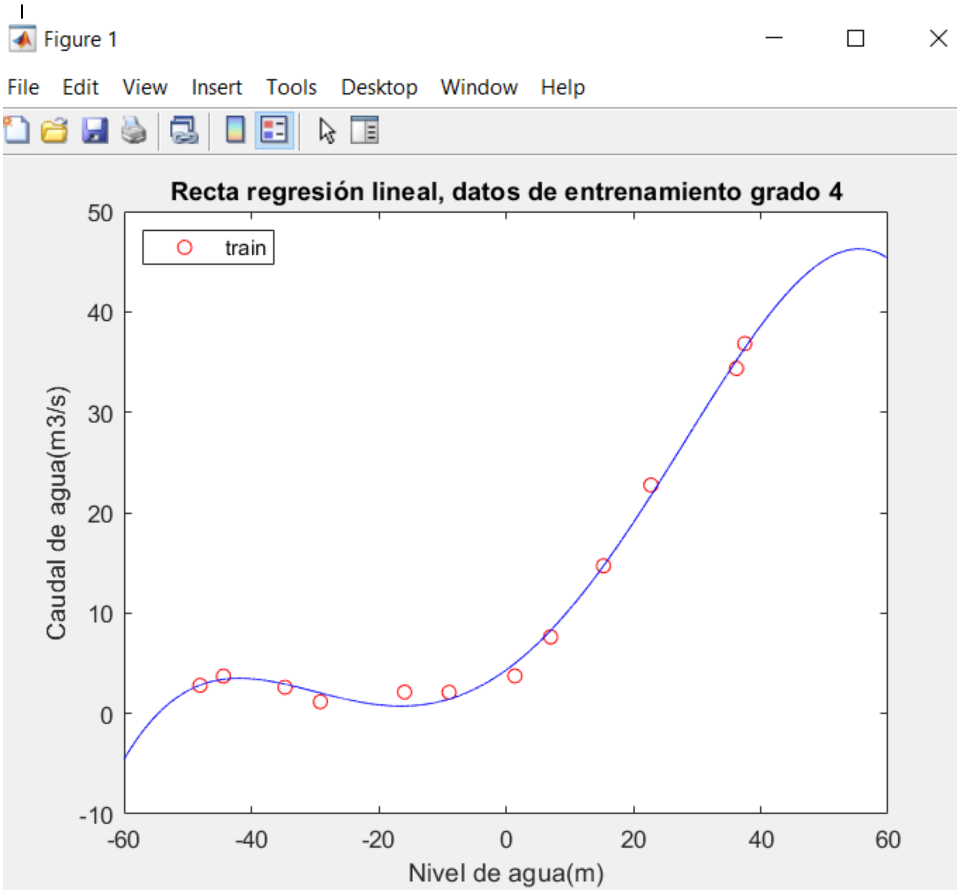


Con la regresión lineal y entrenamiento de grado 3 nos queda una ecuación como la siguiente: y=thetha0*x^3+theta1*x^2+theta2*x+theta3

Entrenamiento grado 4

```
>> polymodel4=polyfitn(xtrain,ytrain,4)

polymodel4 =

  struct with fields:

      ModelTerms: [5×1 double]
    Coefficients: [-2.9924e-06 -1.1756e-05 0.0152 0.4581 4.3286]
    ParameterVar: [9.4201e-13 8.1059e-10 3.0135e-06 0.0012 0.3172]
    ParameterStd: [9.7057e-07 2.8471e-05 0.0017 0.0353 0.5632]
             DoF: 7
               p: [0.0177 0.6920 5.0380e-05 3.7668e-06 1.1762e-04]
              R2: 0.9961
      AdjustedR2: 0.9939
            RMSE: 0.7795
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred4 = polyvaln(polymodel4,x);
plot(xtrain,ytrain,'ro',x,ypred4,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 4');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

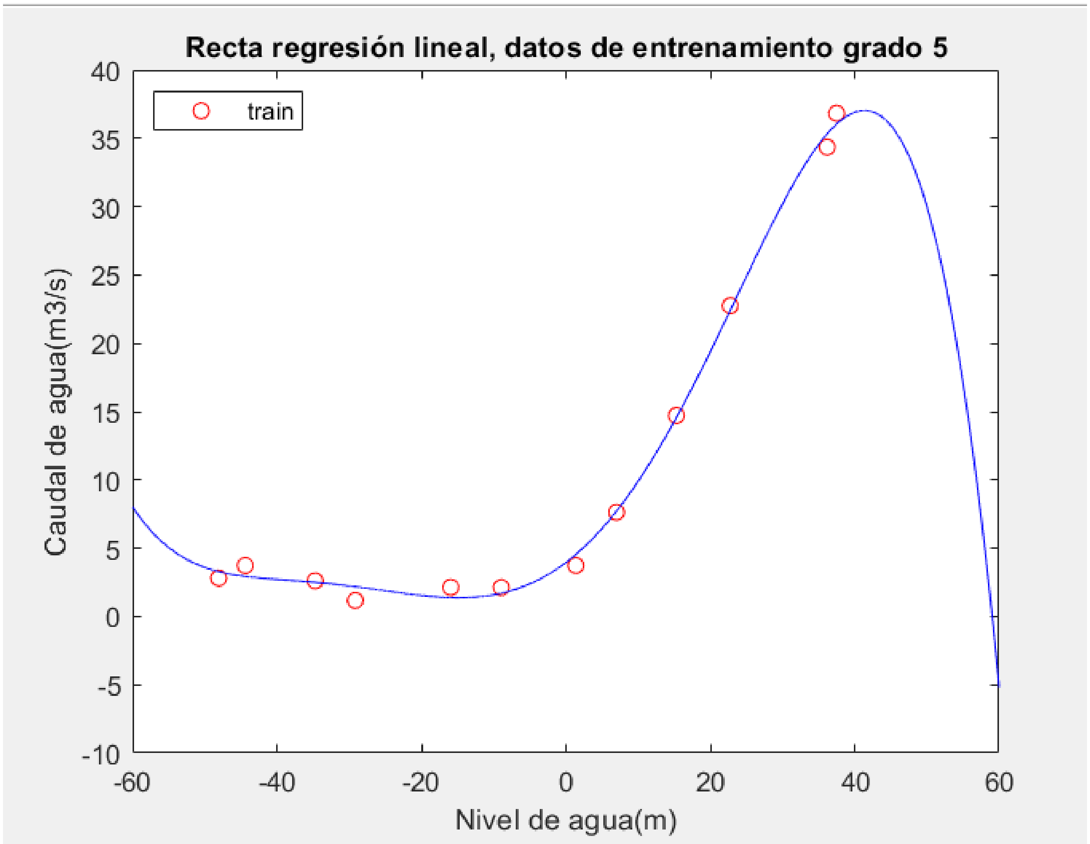


Con la regresión lineal y entrenamiento de grado 4 nos queda una ecuación como la siguiente: y=thetha0*x^4+theta1*x^3+theta2*x^2+theta3*x+theta4

Entrenamiento grado 5

```
>> polymodel5=polyfitn(xtrain,ytrain,5)

polymodel5 =

  struct with fields:

      ModelTerms: [6×1 double]
    Coefficients: [-7.9201e-08 -5.3050e-06 1.4302e-04 0.0184 … ]
    ParameterVar: [2.2869e-15 2.7040e-12 9.3821e-09 6.0771e-06 … ]
    ParameterStd: [4.7822e-08 1.6444e-06 9.6861e-05 0.0025 … ]
             DoF: 6
               p: [0.1488 0.0180 0.1903 2.9848e-04 1.3519e-04 … ]
              R2: 0.9973
      AdjustedR2: 0.9951
            RMSE: 0.6457
        VarNames: {'X1'}
```

```
> x=(-60:0.01:60);
> ypred5 = polyvaln(polymodel5,x);
> plot(xtrain,ytrain,'ro',x,ypred5,'b-')
> title('Recta regresión lineal, datos de entrenamiento grado 5');
> legend("train",'Location','northwest','Orientation','vertical');
> xlabel('Nivel de agua(m)');
> ylabel('Caudal de agua(m3/s)');
```

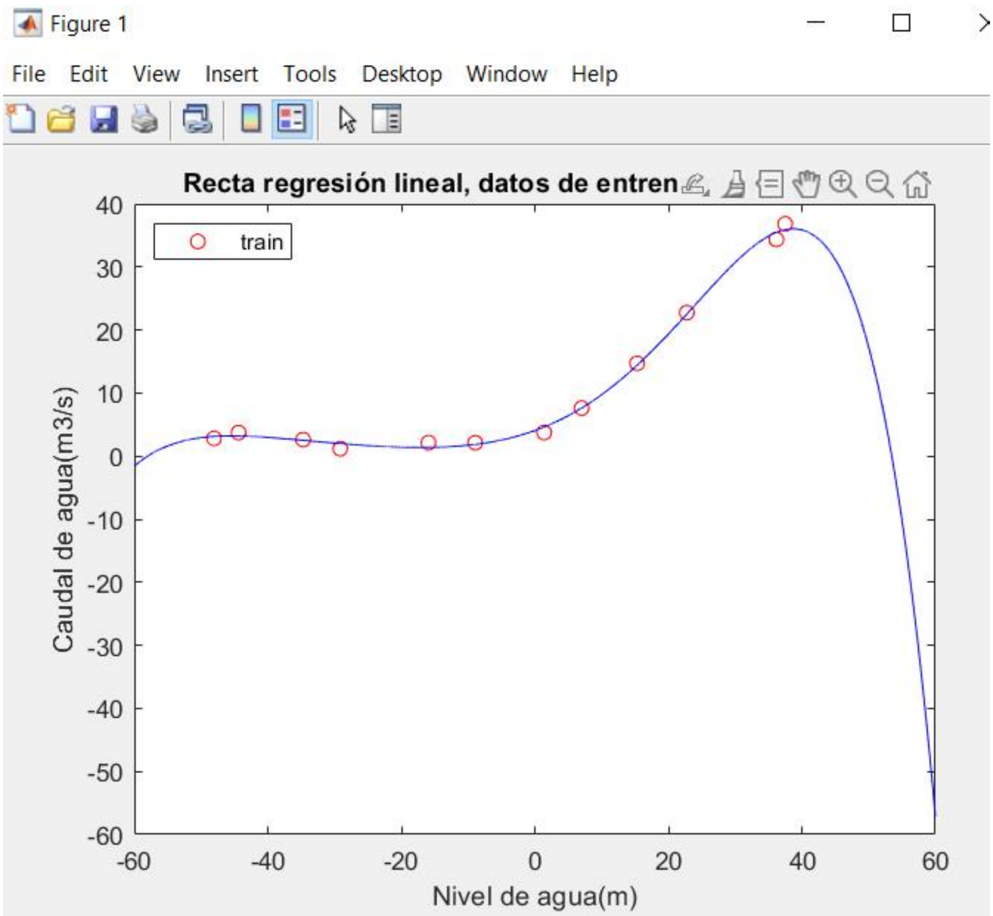


Con la regresión lineal y entrenamiento de grado 5 nos queda una ecuación como la siguiente: y=thetha0*x^5+theta1*x^4+theta2*x^3+theta3*x^2+theta4*x+theta5

Entrenamiento grado 6

```
>> polymodel6=polyfitn(xtrain,ytrain,6)

polymodel6 =

  struct with fields:

      ModelTerms: [7×1 double]
    Coefficients: [-1.3291e-09 -1.2691e-07 -2.5291e-06 … ]
    ParameterVar: [6.7005e-18 1.1239e-14 3.2309e-11 3.4909e-08 … ]
    ParameterStd: [2.5885e-09 1.0601e-07 5.6841e-06 1.8684e-04 … ]
             DoF: 5
               p: [0.6295 0.2849 0.6750 0.2864 0.0059 0.0022 … ]
              R2: 0.9975
      AdjustedR2: 0.9944
            RMSE: 0.6294
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred6 = polyvaln(polymodel6,x);
plot(xtrain,ytrain,'ro',x,ypred6,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 6');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

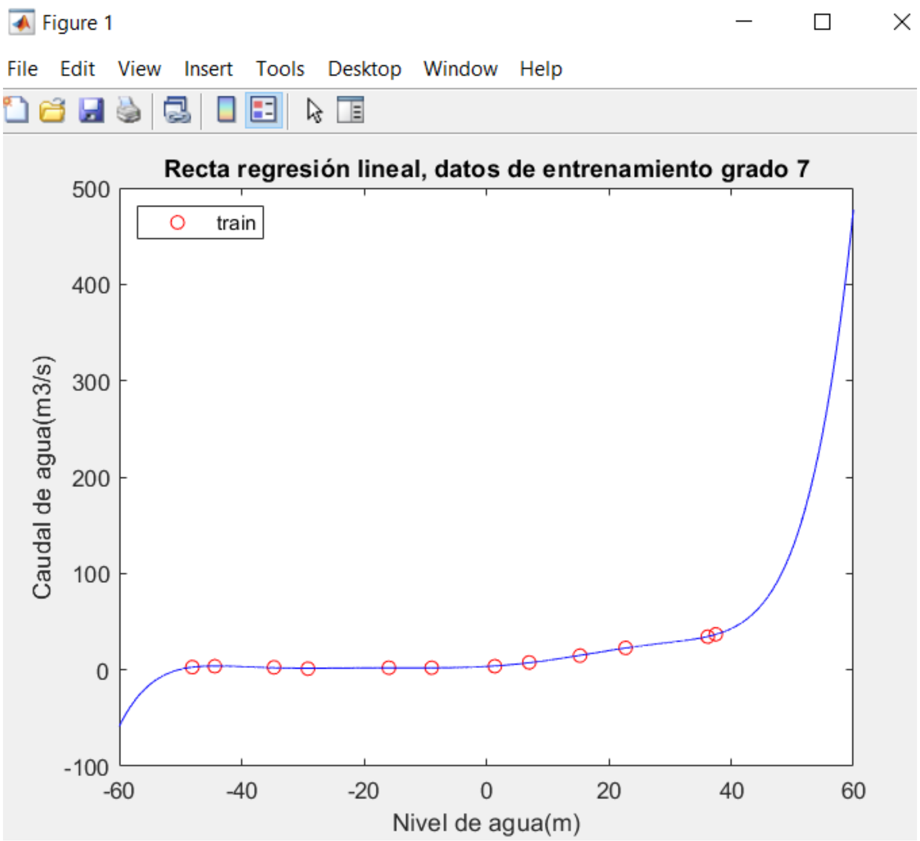


Con la regresión lineal y entrenamiento de grado 6 nos queda una ecuación como la siguiente: y=thetha0*x^6+theta1*x^5+theta2*x^4+theta3*x^3+theta4*x^2+theta5*x+theta6

Entrenamiento grado 7

```
>> polymodel7=polyfitn(xtrain,ytrain,7)

polymodel7 =

  struct with fields:

      ModelTerms: [8×1 double]
    Coefficients: [2.0501e-10 8.7134e-09 -5.3850e-07 ... ]
    ParameterVar: [2.5748e-21 7.8266e-18 1.3143e-14 3.3091e-11 ... ]
    ParameterStd: [5.0742e-11 2.7976e-09 1.1464e-07 5.7525e-06 ... ]
             DoF: 4
               p: [0.0156 0.0357 0.0093 0.0167 0.0156 ... ]
              R2: 0.9995
      AdjustedR2: 0.9986
            RMSE: 0.2792
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred7 = polyvaln(polymodel7,x);
plot(xtrain,ytrain,'ro',x,ypred7,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 7');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

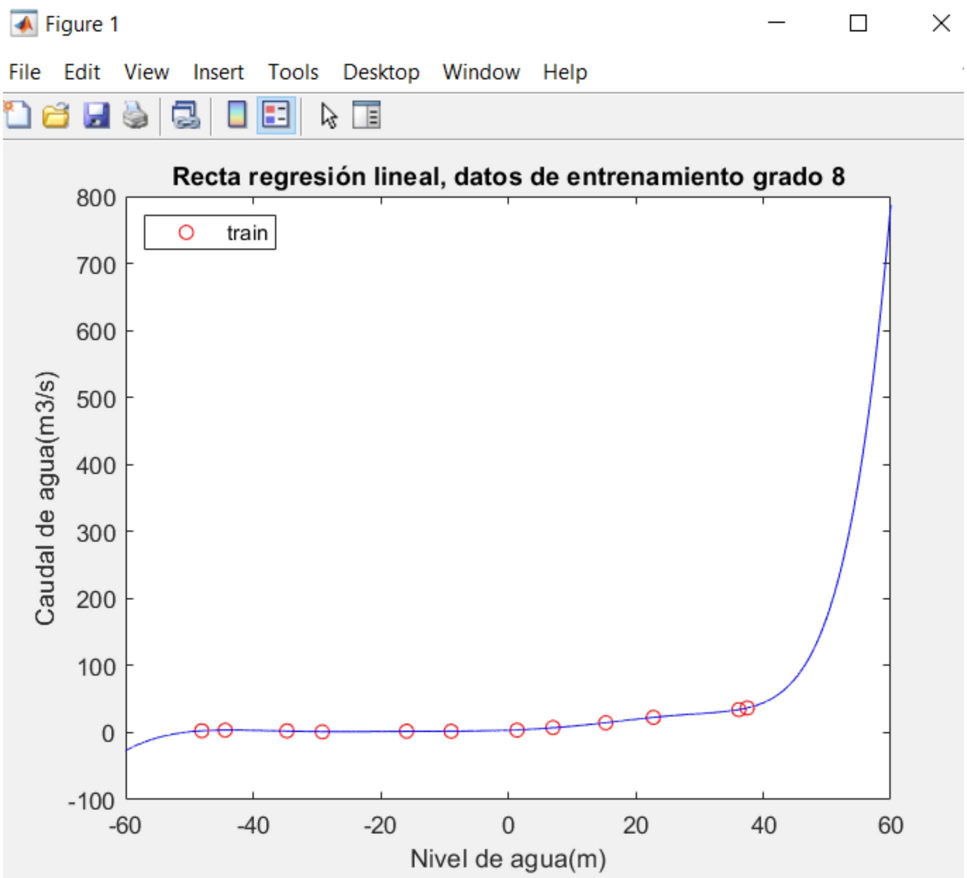


Con la regresión lineal y entrenamiento de grado 7 nos queda una ecuación como la siguiente: y=thetha0*x^7+theta1*x^6+theta2*x^5+theta3*x^4+theta4*x^3+theta5*x^2+theta6*x+theta7

Entrenamiento grado 8

```
>> polymodel8=polyfitn(xtrain,ytrain,8)

polymodel8 =

  struct with fields:

      ModelTerms: [9×1 double]
    Coefficients: [2.4809e-12 3.2023e-10 1.8355e-09 ... ]
    ParameterVar: [5.8747e-24 1.5216e-20 5.2888e-17 9.2987e-14 ... ]
    ParameterStd: [2.4238e-12 1.2335e-10 7.2724e-09 3.0494e-07 ... ]
             DoF: 3
               p: [0.3814 0.0806 0.8170 0.0728 0.1482 0.0673 ... ]
              R2: 0.9996
      AdjustedR2: 0.9986
            RMSE: 0.2404
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred8 = polyvaln(polymodel8,x);
plot(xtrain,ytrain,'ro',x,ypred8,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 8');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

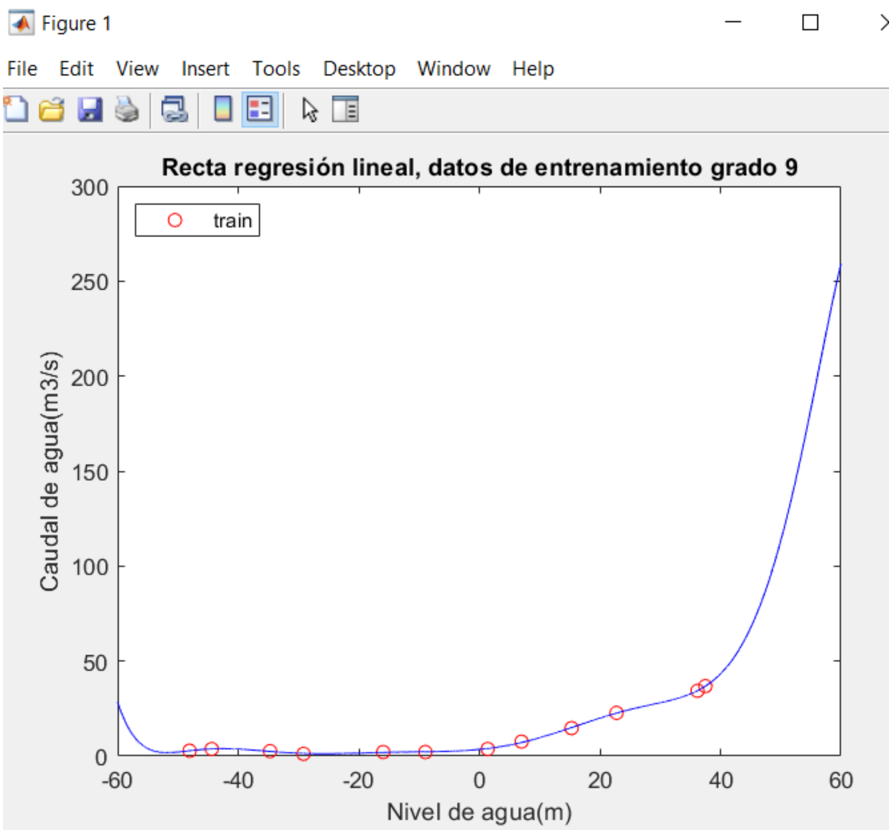


Con la regresión lineal y entrenamiento de grado 8 nos queda una ecuación como la siguiente: y=thetha0*x^8+theta1*x^7+theta2*x^6+theta3*x^5+theta4*x^4+theta5*x^3+theta6*x^2+theta7*x+theta8

Entrenamiento grado 9

```
>> polymodel9=polyfitn(xtrain,ytrain,9)

polymodel9 =

  struct with fields:

      ModelTerms: [10×1 double]
    Coefficients: [-8.4579e-14 -1.3687e-12 5.9500e-10 … ]
    ParameterVar: [1.6559e-26 4.1549e-23 1.9352e-19 3.6853e-16 … ]
    ParameterStd: [1.2868e-13 6.4459e-12 4.3991e-10 1.9197e-08 … ]
             DoF: 2
               p: [0.5785 0.8515 0.3088 0.5605 0.1858 0.2814 … ]
              R2: 0.9997
      AdjustedR2: 0.9983
            RMSE: 0.2180
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred9 = polyvaln(polymodel9,x);
plot(xtrain,ytrain,'ro',x,ypred9,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 9');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

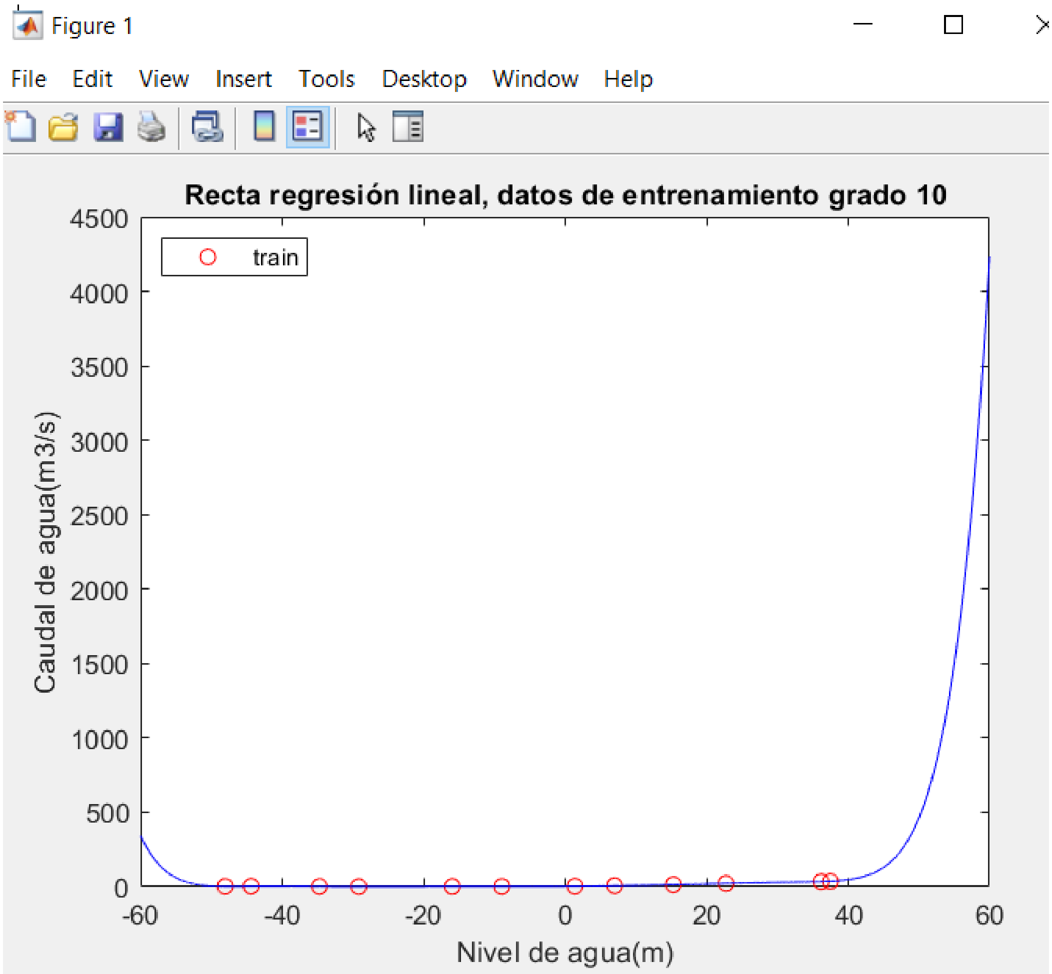


Con la regresión lineal y entrenamiento de grado 9 nos queda una ecuación como la siguiente: y=thetha0*x^9+theta1*x^8+theta2*x^7+theta3*x^6+theta4*x^5+theta5*x^4+theta6*x^3+theta7*x^2+theta8*x+theta9

Entrenamiento grado 10

```
>> polymodel10=polyfitn(xtrain,ytrain,10)

polymodel10 = 

  struct with fields:

      ModelTerms: [11×1 double]
    Coefficients: [1.0873e-14 4.8992e-13 -3.7827e-11 … ]
    ParameterVar: [2.7724e-29 8.3685e-26 3.2748e-22 9.6625e-19 … ]
    ParameterStd: [5.2653e-15 2.8928e-13 1.8096e-11 9.8298e-10 … ]
             DoF: 1
               p: [0.2871 0.3396 0.2841 0.3993 0.2563 0.5211 … ]
              R2: 0.9999
      AdjustedR2: 0.9994
            RMSE: 0.0950
        VarNames: {'X1'}
```

```
x=(-60:0.01:60);
ypred10 = polyvaln(polymodel10,x);
plot(xtrain,ytrain,'ro',x,ypred10,'b-')
title('Recta regresión lineal, datos de entrenamiento grado 10');
legend("train",'Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```



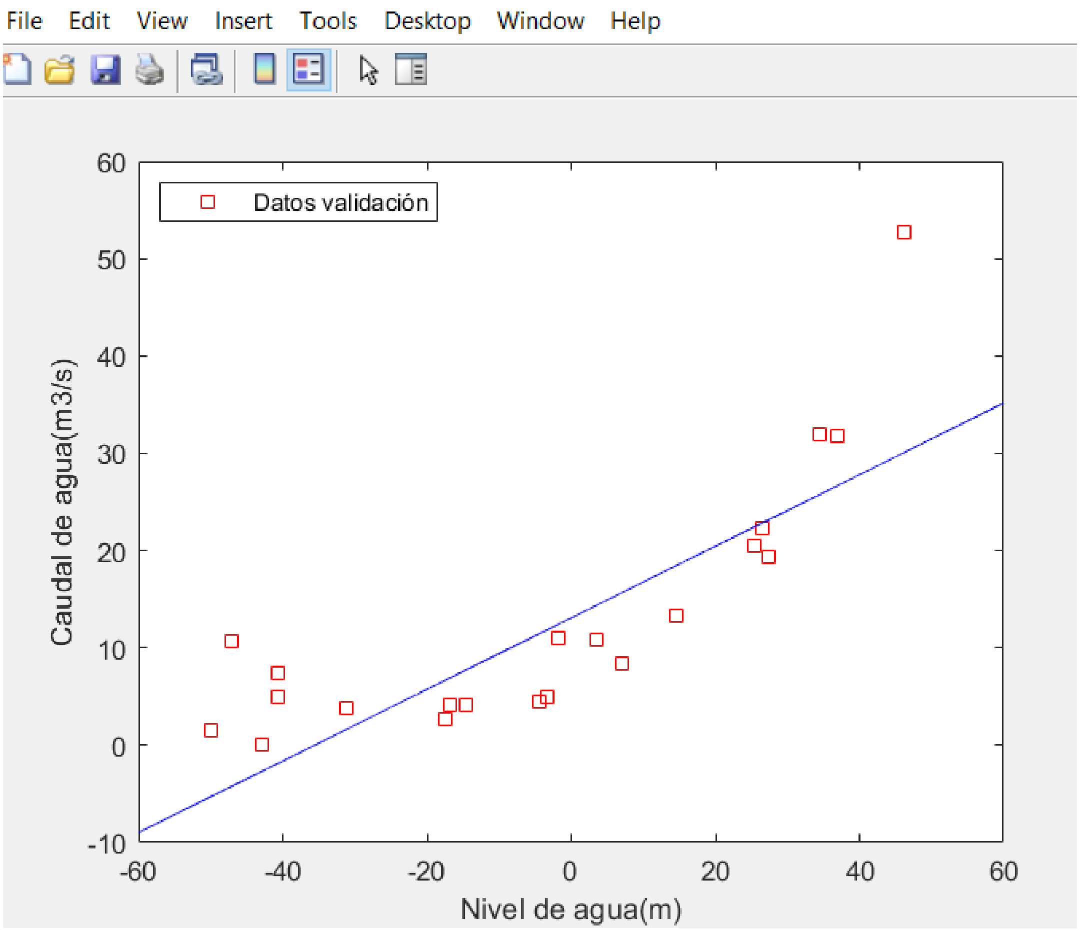
Con la regresión lineal y entrenamiento de grado 10 nos queda una ecuación como la siguiente: y=thetha0*x^10+theta1*x^9+theta2*x^8+theta3*x^7+theta4*x^6+theta5*x^5+theta6*x^4+theta7*x^3+ theta8*x^2+theta9*x+theta10

1.3.5 ELECCIÓN DEL MODELO DE LA REGRESIÓN POLINOMINAL.

El mejor modelo es el de validación de grado 3 porque nos sale el R^2 mayor y por lo tanto es el que mejor se ajusta.

Validación grado 1

```
plot(xval,yval,'rs',x,polyvaln(polymodel1,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

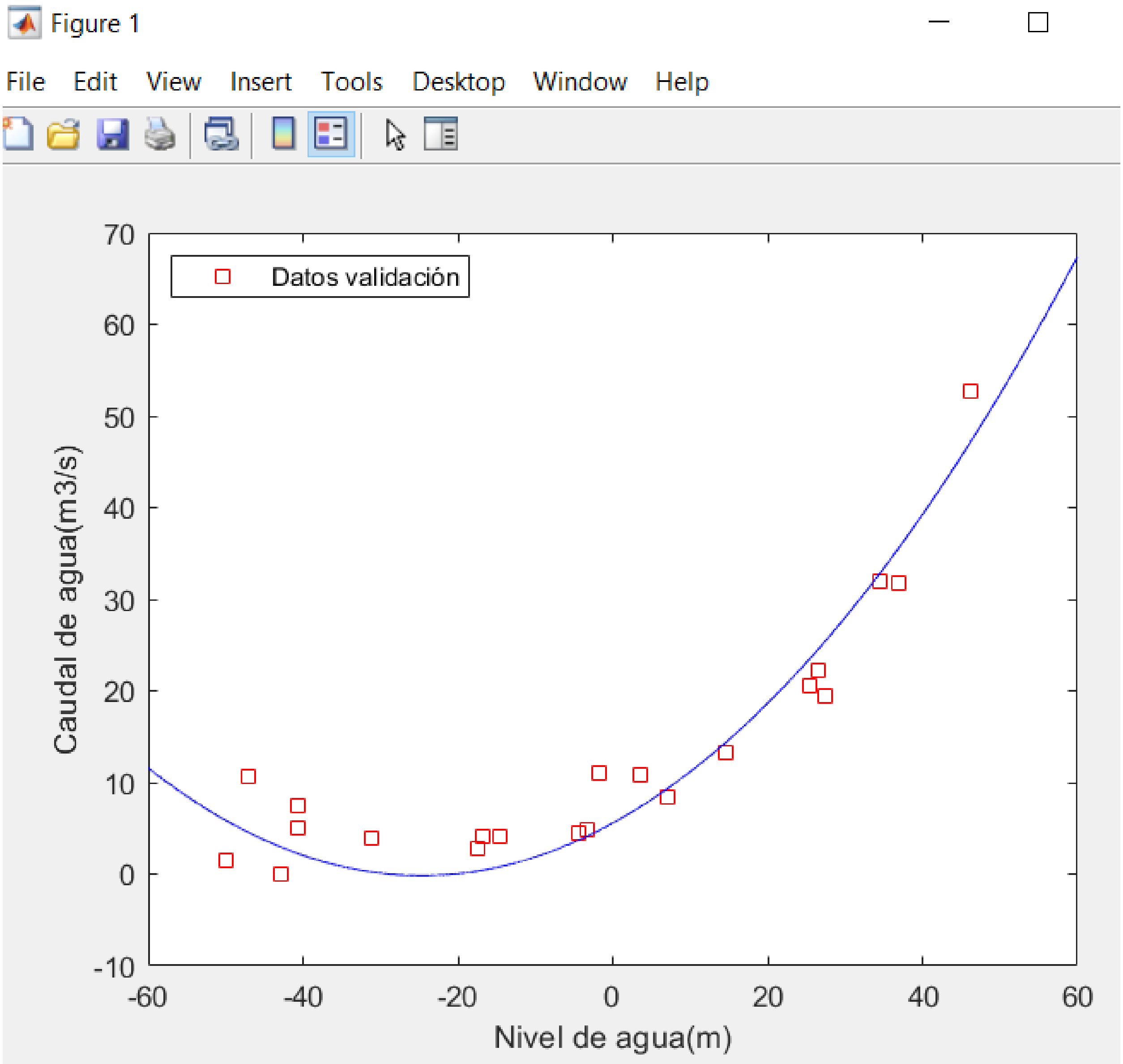



```
>> yvalpred=polyvaln(polymodel1,xval);
>> R2(1)=calculateR2(yval,yvalpred)

R2 =

    0.6358
```

Con la regresión lineal y entrenamiento de grado 1 nos queda una ecuación como la siguiente: y=thetha0*x+theta1

Validación grado 2

```
plot(xval,yval,'rs',x,polyvaln(polymodel2,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

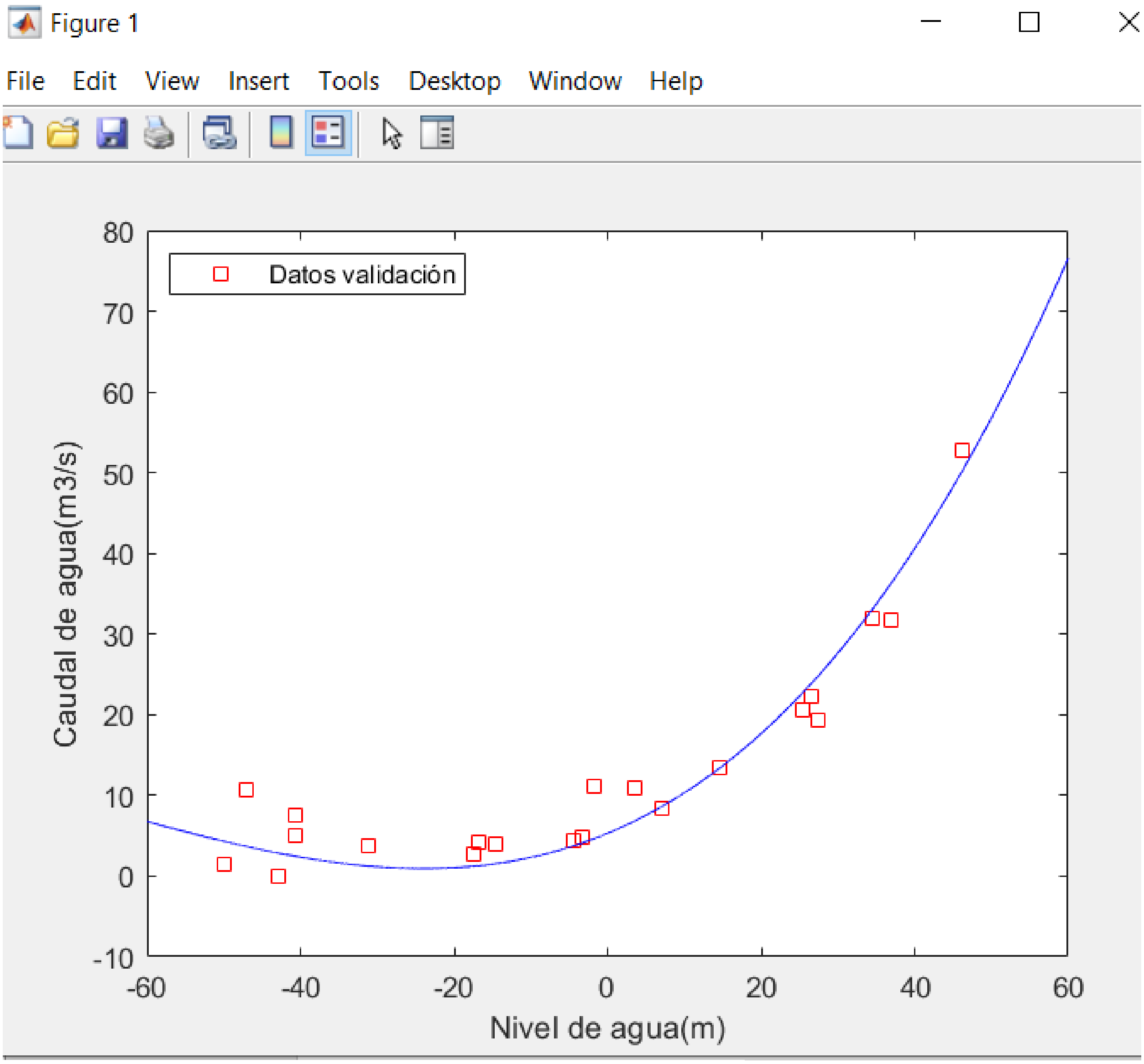



```
>> yvalpred=polyvaln(polymodel2,xval);
>> R2(2)=calculateR2(yval,yvalpred)

R2 =

    0.6358    0.9135
```

Con la regresión lineal y entrenamiento de grado 2 nos queda una ecuación como la siguiente: y=thetha0*x^2+theta1*x+theta2

Validación grado 3

```
plot(xval,yval,'rs',x,polyvaln(polymodel3,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

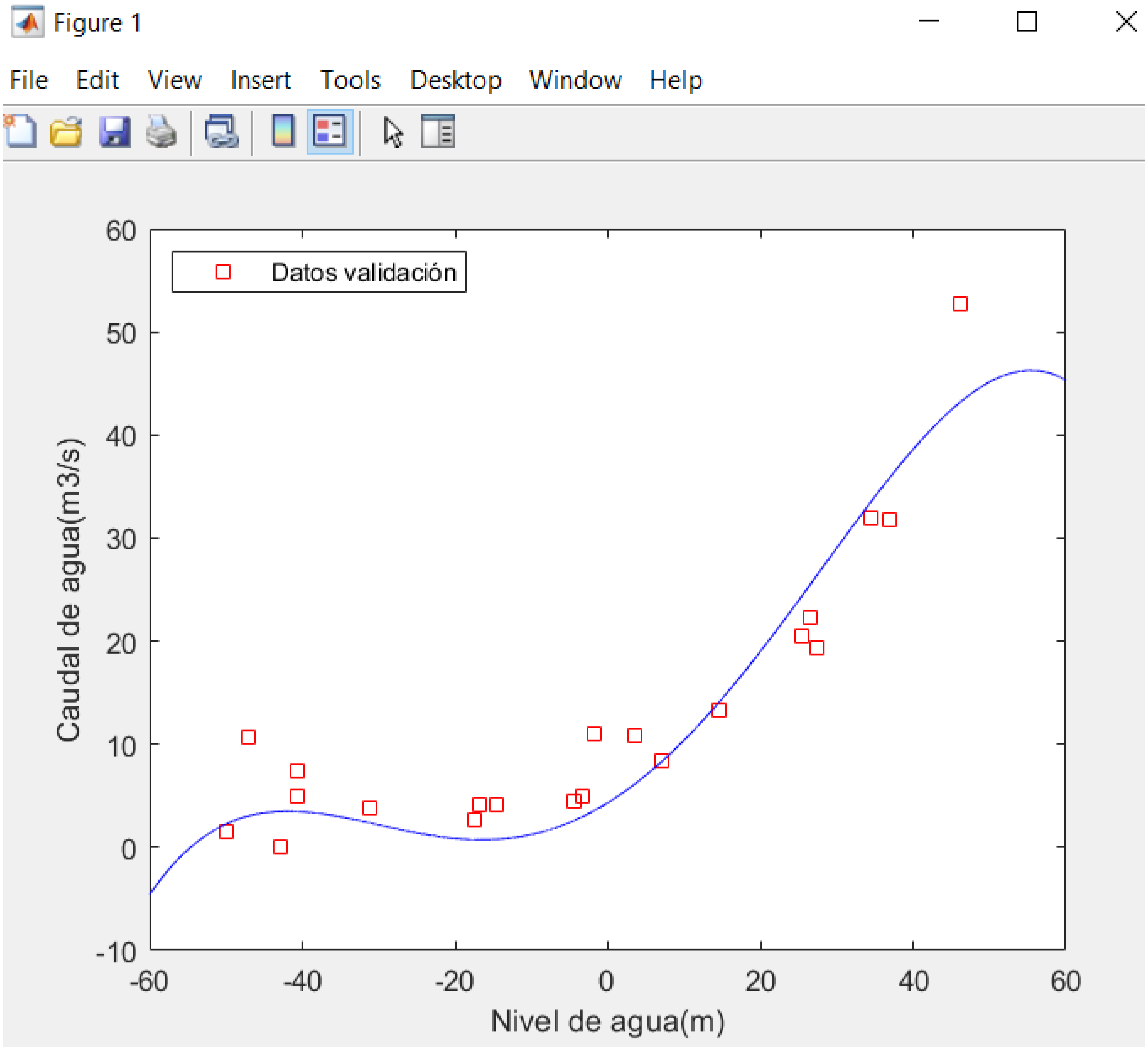



```
>> yvalpred=polyvaln(polymodel3,xval);
>> R2(3)=calculateR2(yval,yvalpred)

R2 =

    0.6358    0.9135    0.9286
```

Con la regresión lineal y entrenamiento de grado 3 nos queda una ecuación como la siguiente: y=thetha0*x^3+theta1*x^2+theta2*x+theta3

Validación grado 4

```
plot(xval,yval,'rs',x,polyvaln(polymodel4,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

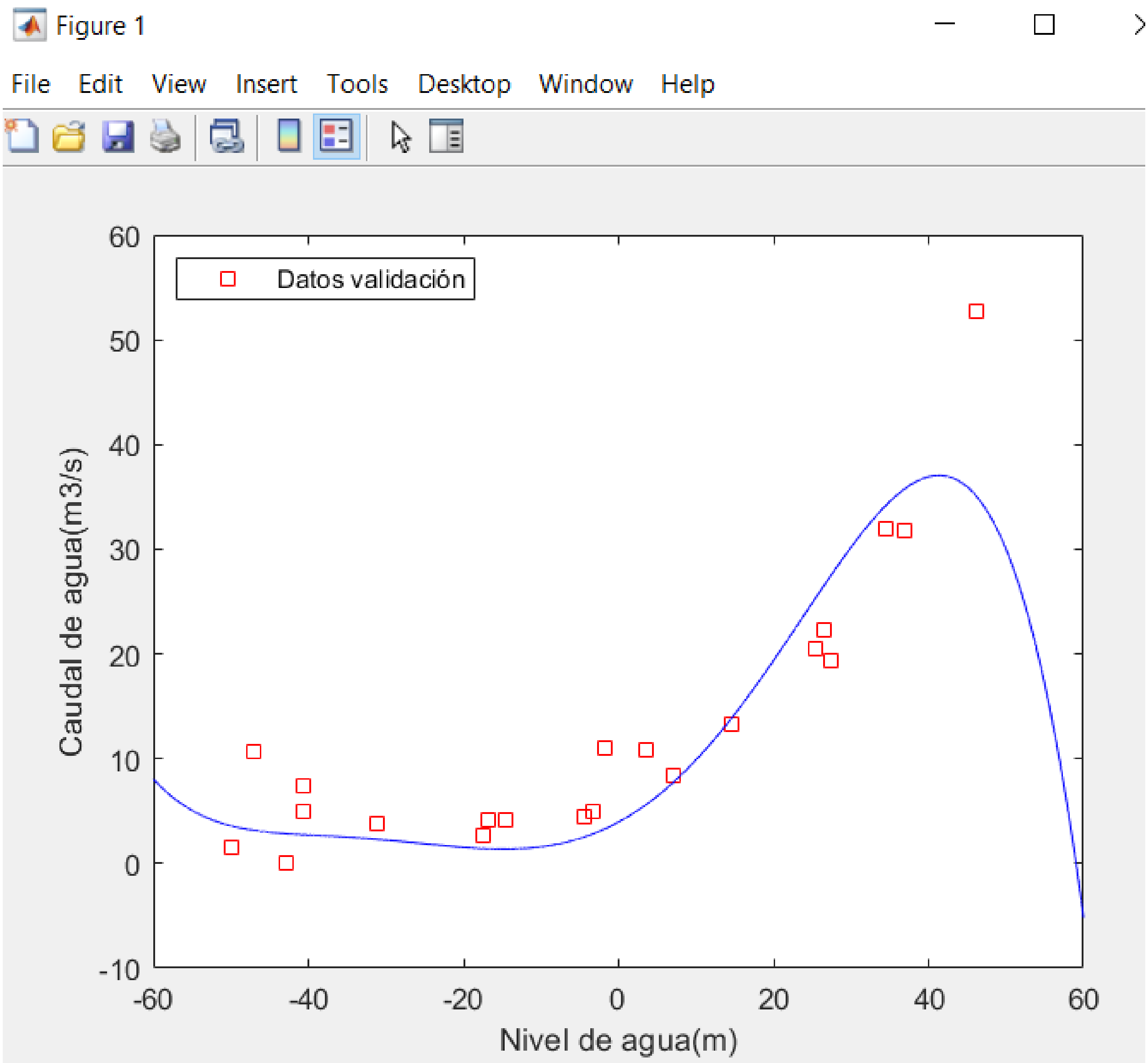



```
>> yvalpred=polyvaln(polymodel4,xval);
>> R2(4)=calculateR2(yval,yvalpred)

R2 =

    0.6358    0.9135    0.9286    0.8860
```

Con la regresión lineal y entrenamiento de grado 4 nos queda una ecuación como la siguiente: y=thetha0*x^4+theta1*x^3+theta2*x^2+theta3*x+theta4

Validación grado 5

```
plot(xval,yval,'rs',x,polyvaln(polymodel5,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

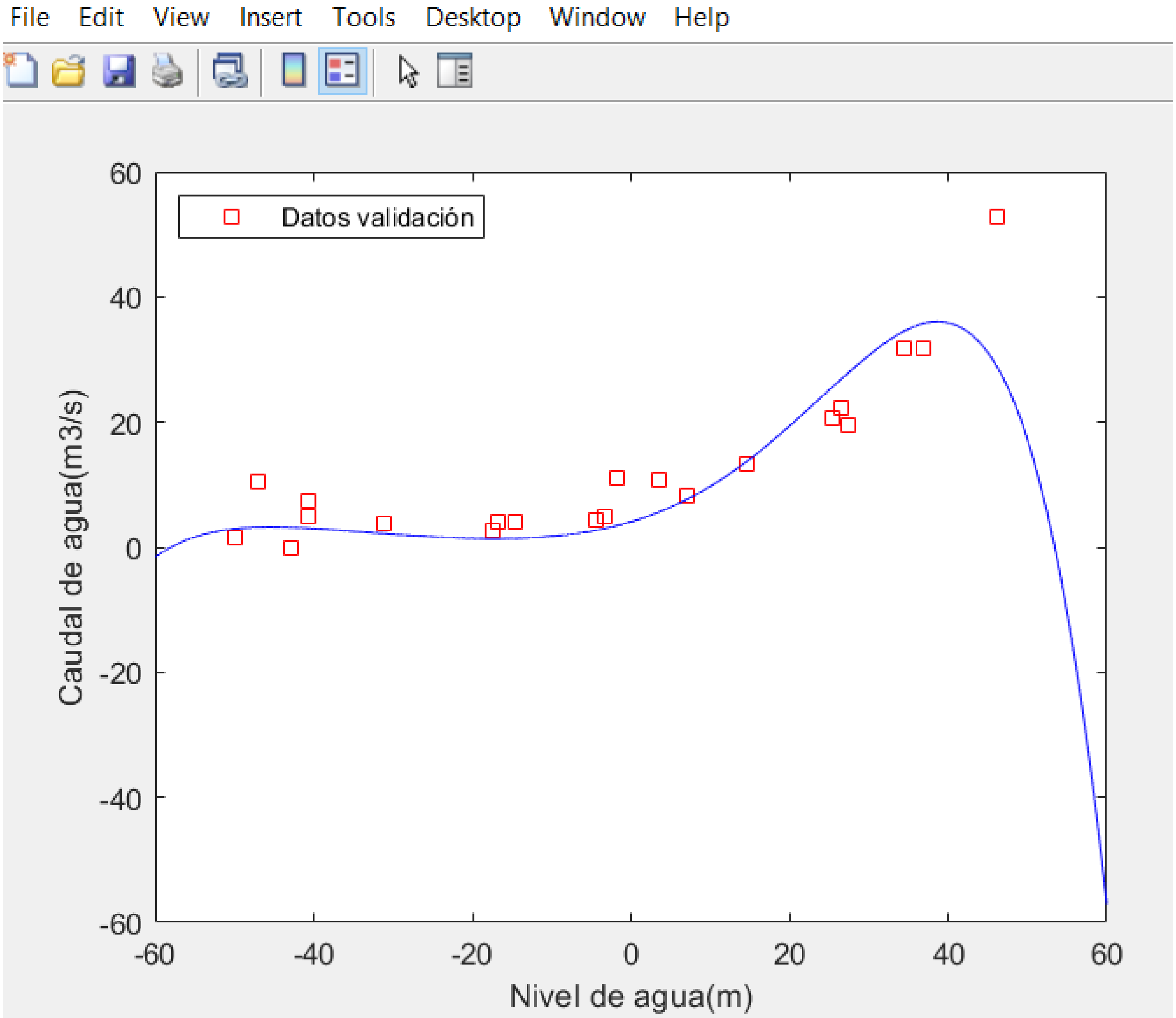



```
>> yvalpred=polyvaln(polymodel5,xval);
>> R2(5)=calculateR2(yval,yvalpred)

R2 =

    0.6358    0.9135    0.9286    0.8860    0.8066
```

Con la regresión lineal y entrenamiento de grado 5 nos queda una ecuación como la siguiente: y=thetha0*x^5+theta1*x^4+theta2*x^3+theta3*x^2+theta4*x+theta5

Validación grado 6

```
plot(xval,yval,'rs',x,polyvaln(polymodel6,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

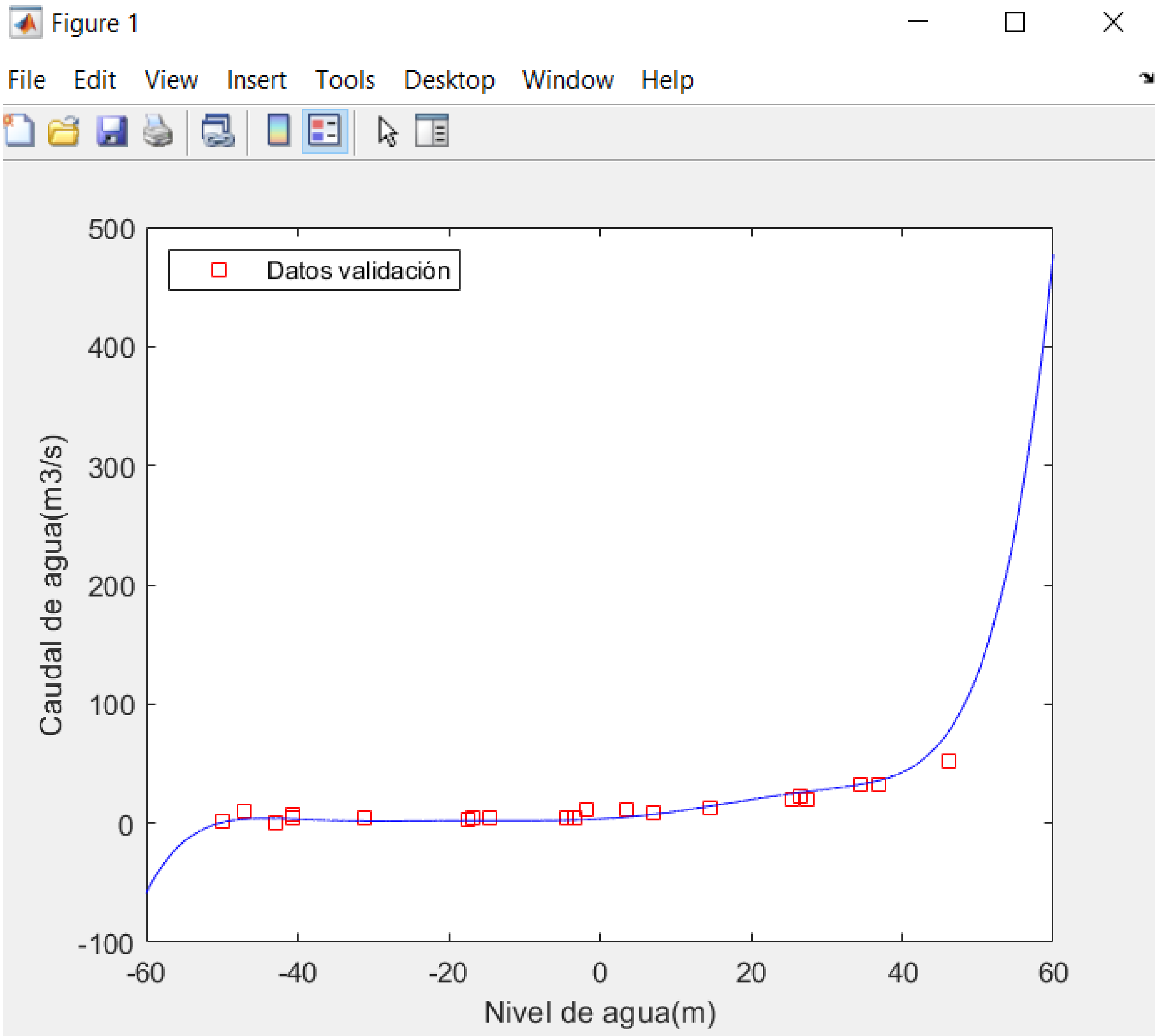



```
>> yvalpred=polyvaln(polymodel6,xval);
>> R2(6)=calculateR2(yval,yvalpred)

R2 =

    0.6358    0.9135    0.9286    0.8860    0.8066    0.7263
```

Con la regresión lineal y entrenamiento de grado 6 nos queda una ecuación como la siguiente: y=thetha0*x^6+theta1*x^5+theta2*x^4+theta3*x^3+theta4*x^2+theta5*x+theta6

Validación grado 7

```
plot(xval,yval,'rs',x,polyvaln(polymodel7,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

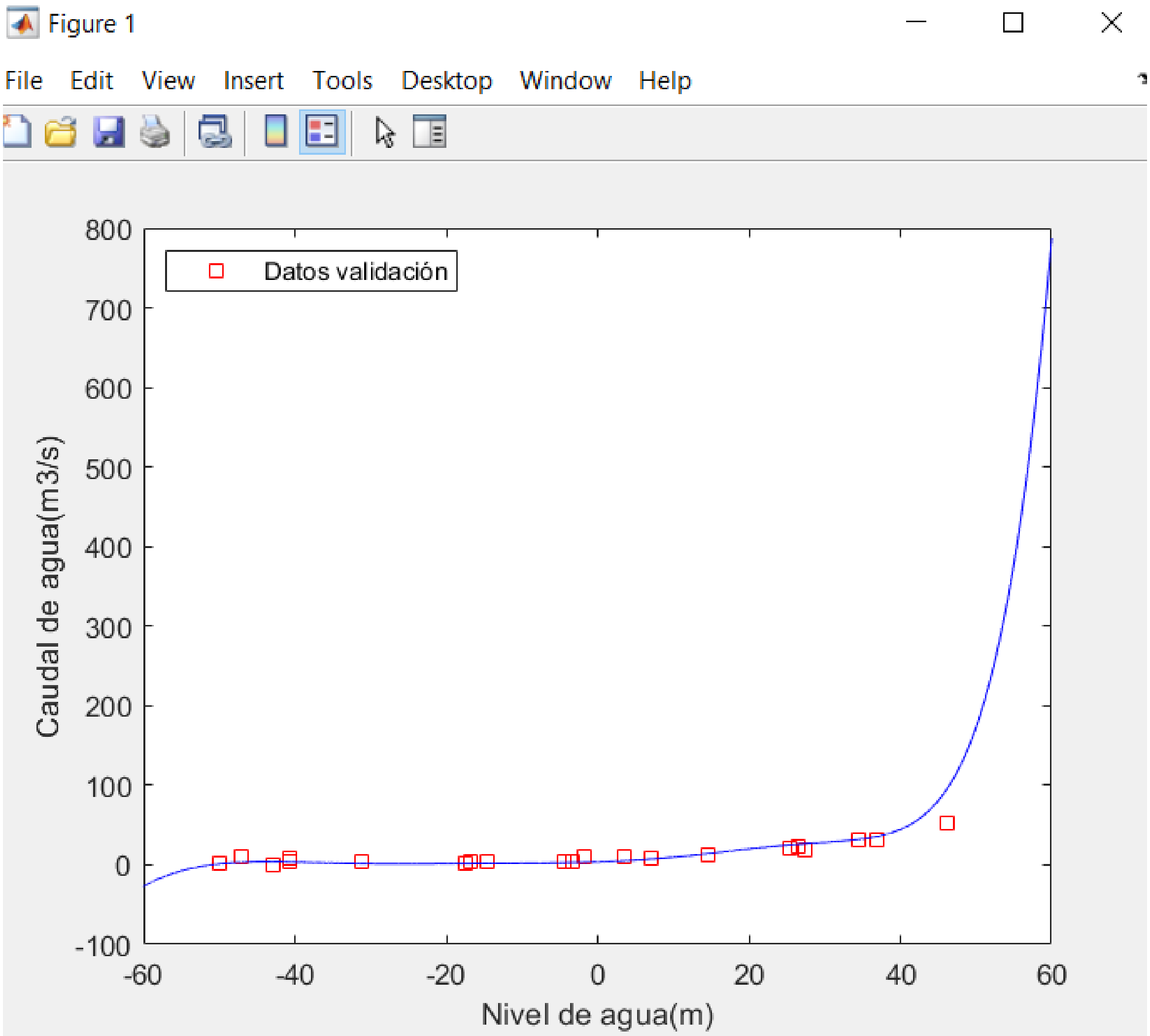



```
>> yvalpred=polyvaln(polymodel7,xval);
>> R2(7)=calculateR2(yval,yvalpred)

R2 =

    0.6358    0.9135    0.9286    0.8860    0.8066    0.7263    0.7170
```

Con la regresión lineal y entrenamiento de grado 7 nos queda una ecuación como la siguiente: y=thetha0*x^7+theta1*x^6+theta2*x^5+theta3*x^4+theta4*x^3+theta5*x^2+theta6*x+theta7

Validación grado 8

```
plot(xval,yval,'rs',x,polyvaln(polymodel8,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

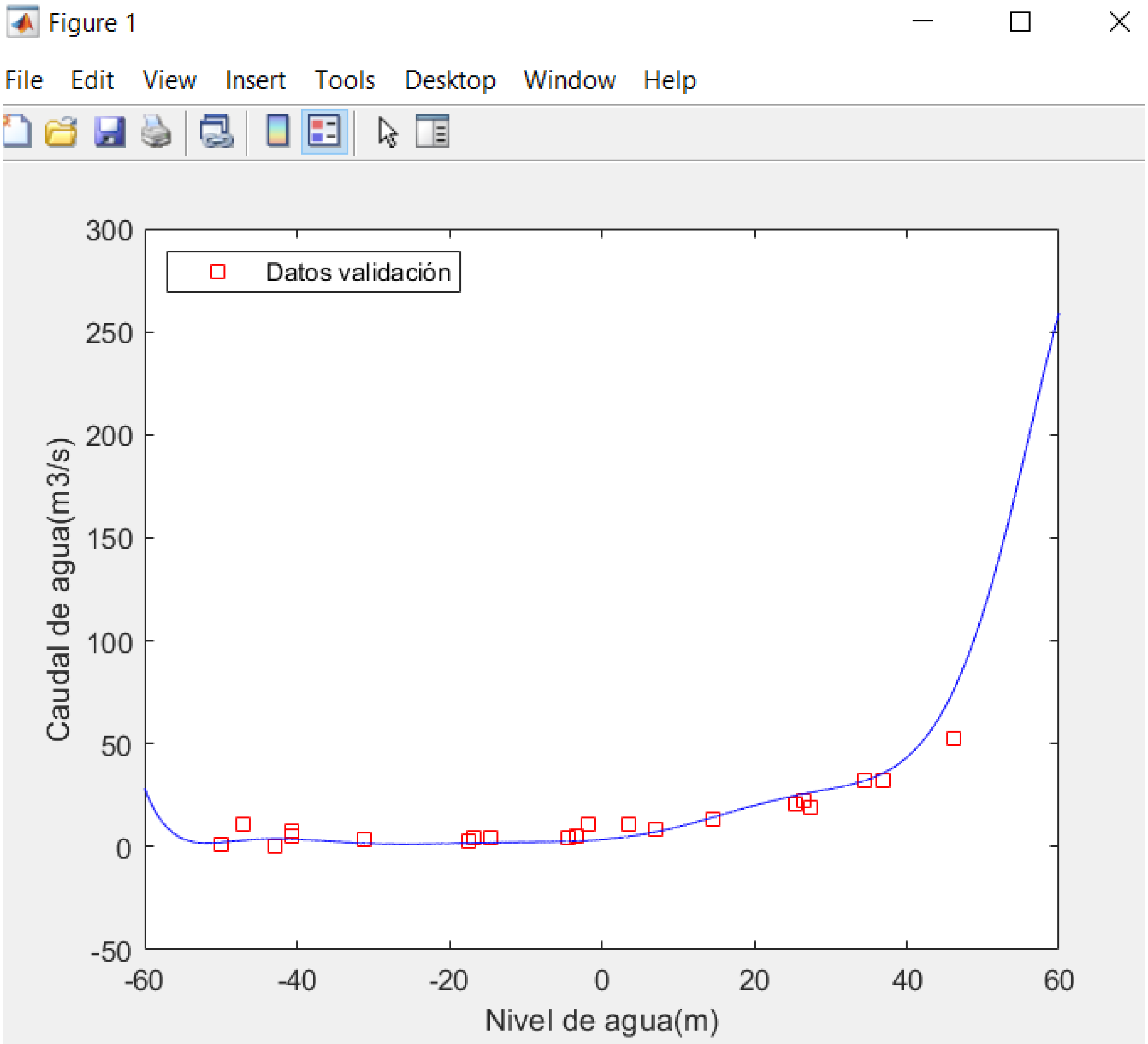



```
>> yvalpred=polyvaln(polymodel8,xval);
>> R2(8)=calculateR2(yval,yvalpred)

R2 =

  Columns 1 through 7

    0.6358    0.9135    0.9286    0.8860    0.8066    0.7263    0.7170

  Column 8

    0.3334
```

Con la regresión lineal y entrenamiento de grado 8 nos queda una ecuación como la siguiente: y=thetha0*x^8+theta1*x^7+theta2*x^6+theta3*x^5+theta4*x^4+theta5*x^3+theta6*x^2+theta7*x+theta8

Validación grado 9

```
plot(xval,yval,'rs',x,polyvaln(polymodel9,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```

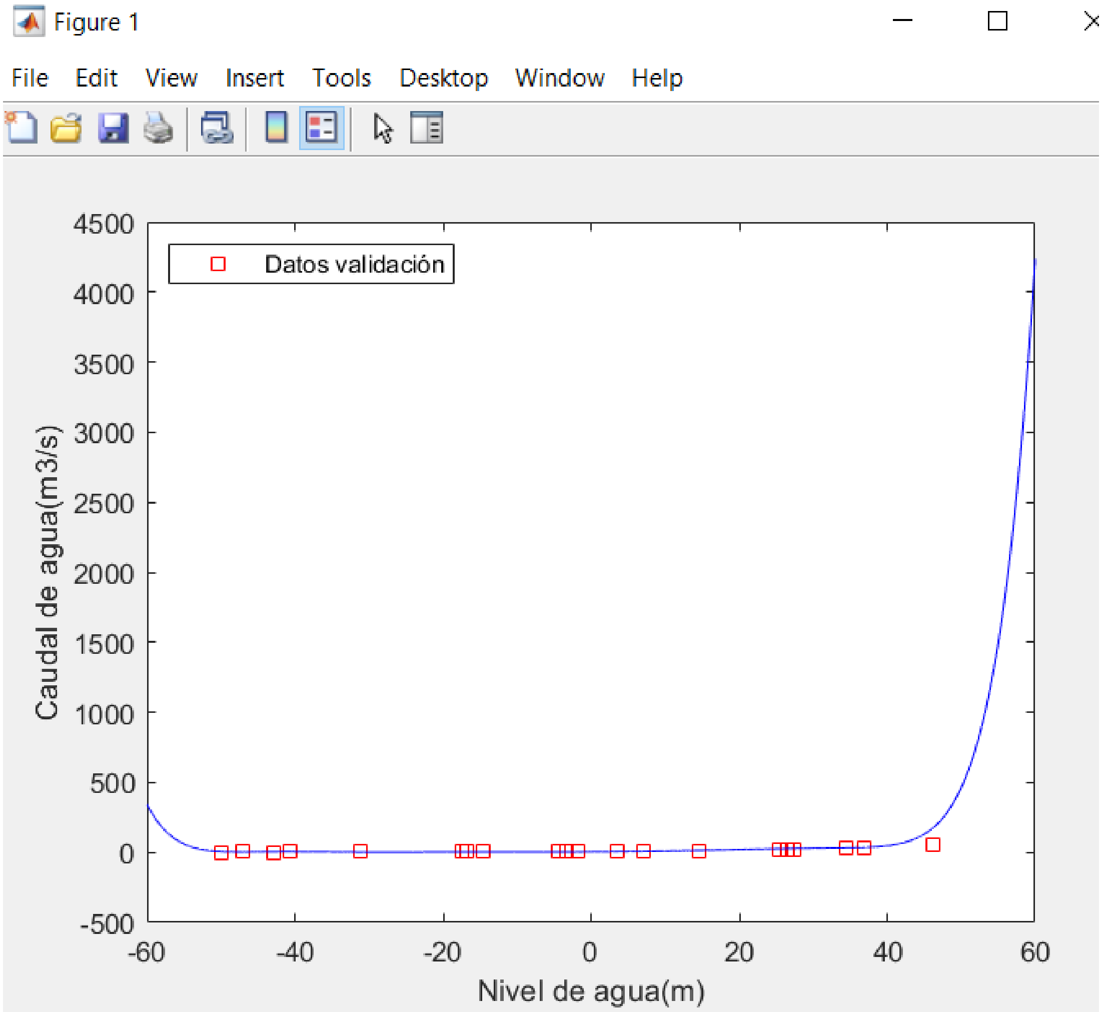



```
>> yvalpred=polyvaln(polymodel9,xval);
>> R2(9)=calculateR2(yval,yvalpred)

R2 =

  Columns 1 through 7

    0.6358    0.9135    0.9286    0.8860    0.8066    0.7263    0.7170

  Columns 8 through 9

    0.3334    0.7387
```

Con la regresión lineal y entrenamiento de grado 9 nos queda una ecuación como la siguiente: y=thetha0*x^9+theta1*x^8+theta2*x^7+theta3*x^6+theta4*x^5+theta5*x^4+theta6*x^3+theta7*x^2+theta8*x+theta9

Validación grado 10

```
plot(xval,yval,'rs',x,polyvaln(polymodel10,x),'b-');
legend('Datos validación','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```



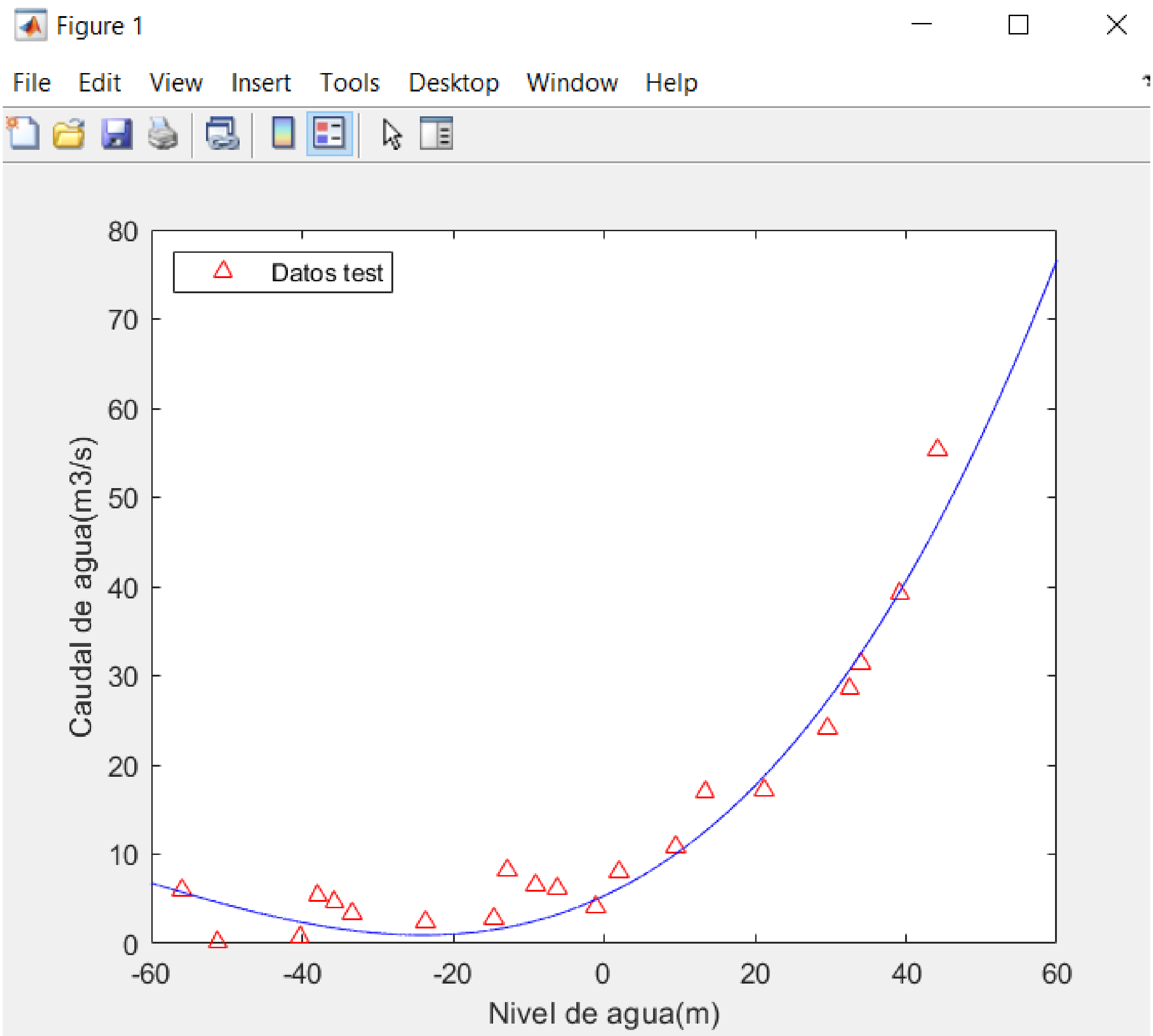

```
>> yvalpred=polyvaln(polymodel10,xval);
>> R2(10)=calculateR2(yval,yvalpred)

R2 =

  Columns 1 through 7

    0.6358    0.9135    0.9286    0.8860    0.8066    0.7263    0.7170

  Columns 8 through 10

    0.3334    0.7387   -3.8200
```

Con la regresión lineal y entrenamiento de grado 10 nos queda una ecuación como la siguiente: y=thetha0*x^10+theta1*x^9+theta2*x^8+theta3*x^7+theta4*x^6+theta5*x^5+theta6*x^4+theta7*x^3+ theta8*x^2+theta9*x+theta10

1.3.6 DESEMPEÑO DE LA REGRESIÓN POLINOMIAL.

El mejor es el de grado 3.

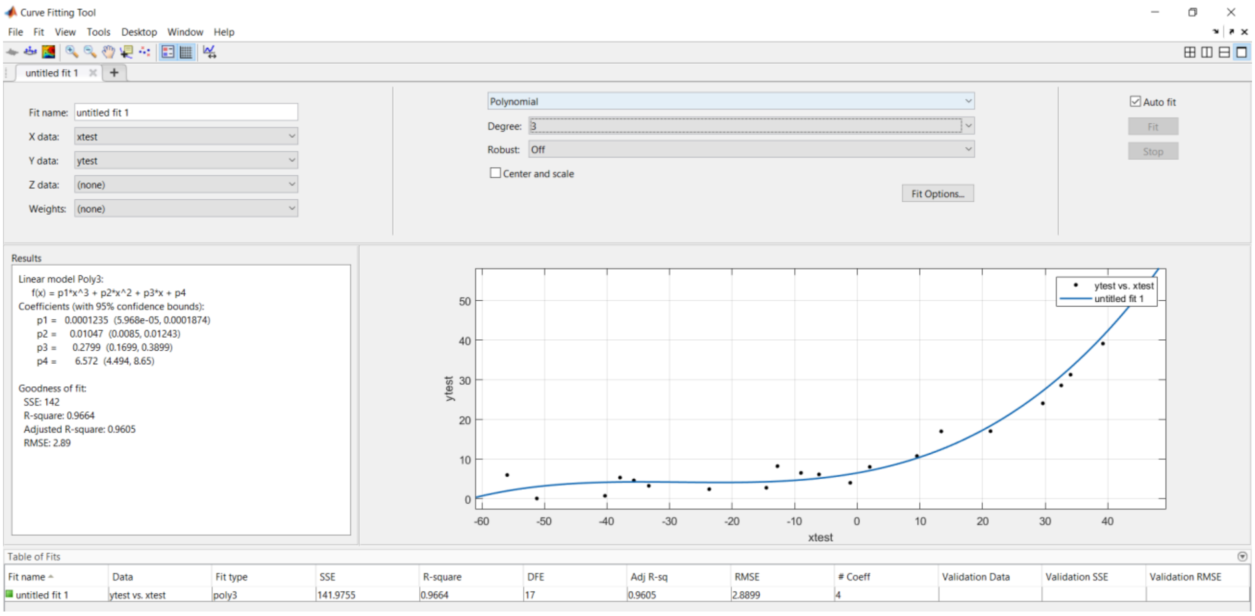
```
plot(xtest,ytest,'r^',x,polyvaln(polymodel3,x),'b-');
legend('Datos test','Location','northwest','Orientation','vertical');
xlabel('Nivel de agua(m)');
ylabel('Caudal de agua(m3/s)');
```




```
>> ytestprend=polyvaln(polymodel3,xtest);
>> R2test=calculateR2(ytest,ytestprend)

R2test =

    0.9454
```

Al escoger el modelo de validación 3 y aplicarlo a los datos de xtest e ytest podemos observar que el desempeño de la regresión polinomial se ajusta bastante bien a los nuevos datos introducidos, ya que R^2 del test tiene un valor del 0.9454, es decir, se ajusta a un 94,54%

1.3.7 APP DE MATLAB.



1.4 CUESTIÓN 4.

1.4.1 INTRODUCCIÓN.

Se implementará una regresión logística para predecir si una máquina fallará en un tiempo inmediato, en base al estado de dos variables que se han controlado durante un largo tiempo y de las cuales depende su correcto funcionamiento, que han dado lugar a una base de datos recogida en el fichero cuestion44_data.xlsx

Por lo tanto, las variables independientes serán la humedad relativa y la presión atmosférica, y la dependiente de las anteriores, será el fallo o no de la máquina en cuestión

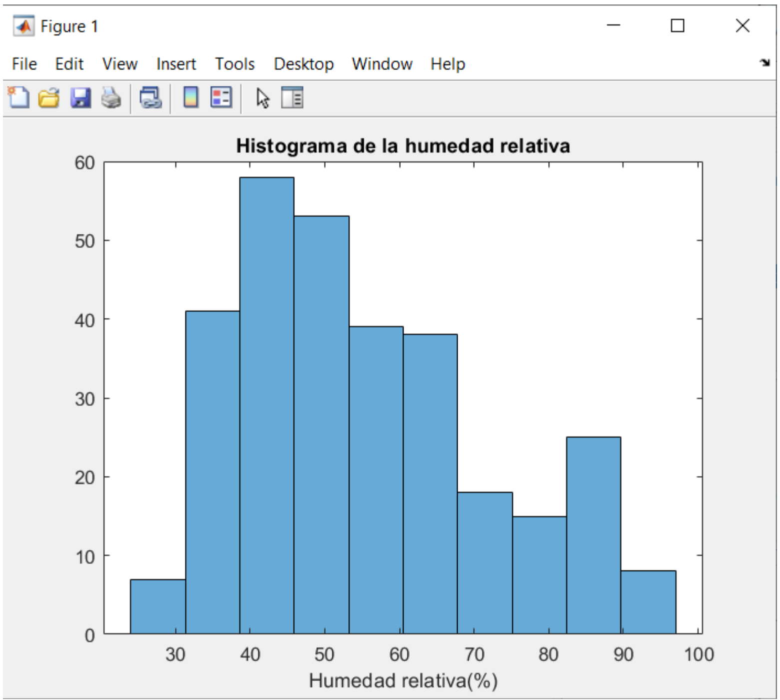
1.4.2 VISUALIZACIÓN DE LOS DATOS.

Carga de datos

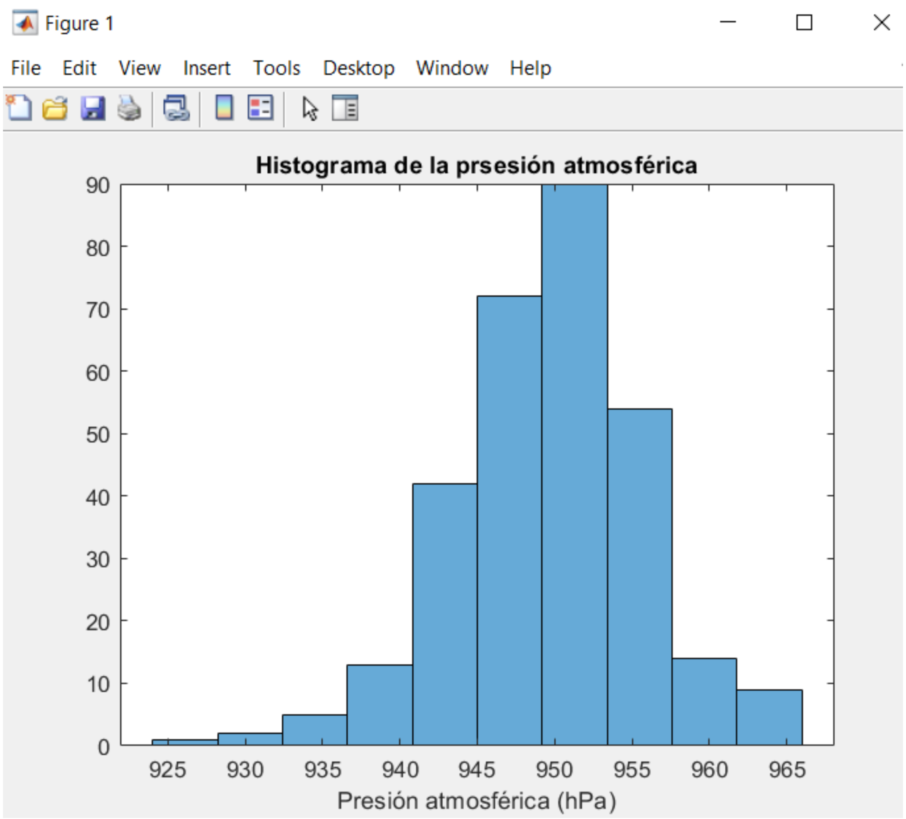
```
datos=readmatrix('cuestion44_data.xlsx');
x=datos(:,1:2);
y=datos(:,3);
y=y+1;
m=length(y);
```

Visualización de datos

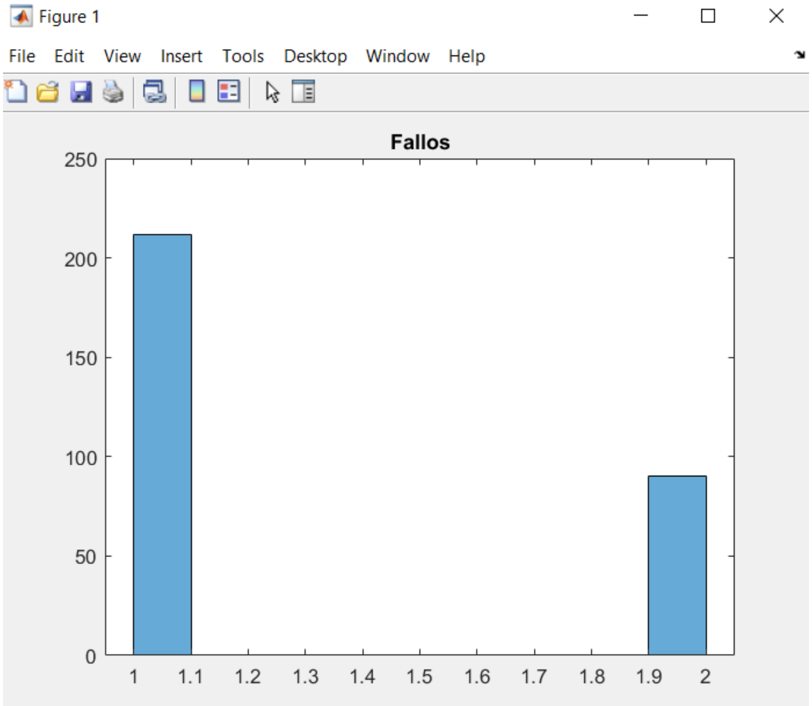
```
histogram(x(:,1),10);
title('Histograma de la humedad relativa');
xlabel('Humedad relativa(%)');
```



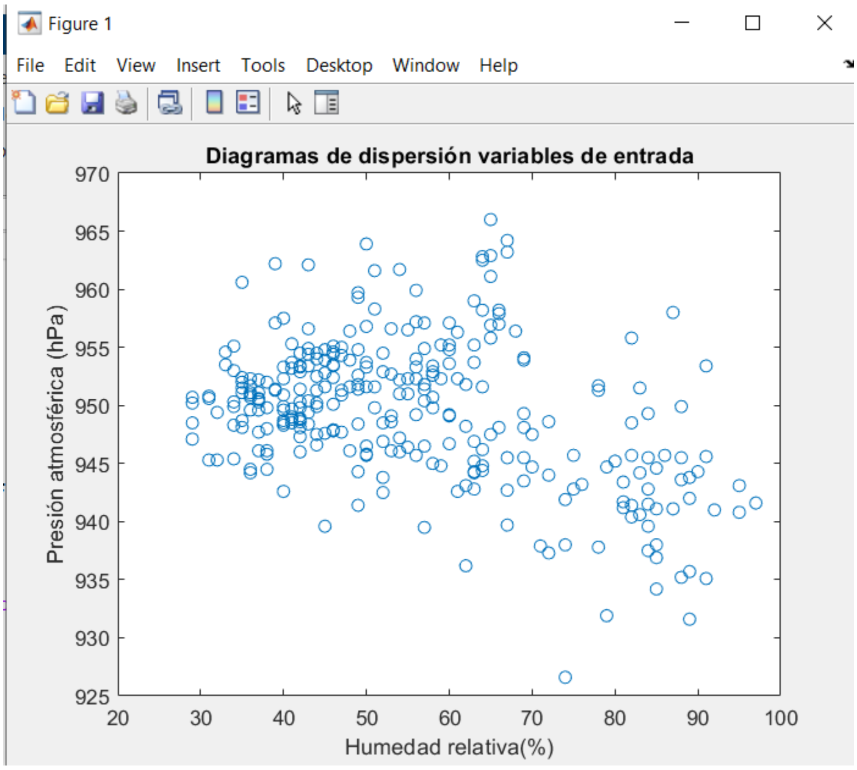

```
histogram(x(:,2),10);
title('Histograma de la prsesión atmosférica');
xlabel('Presión atmosférica (hPa)');
```



```
histogram(y(:),10);
title('Fallos');
```



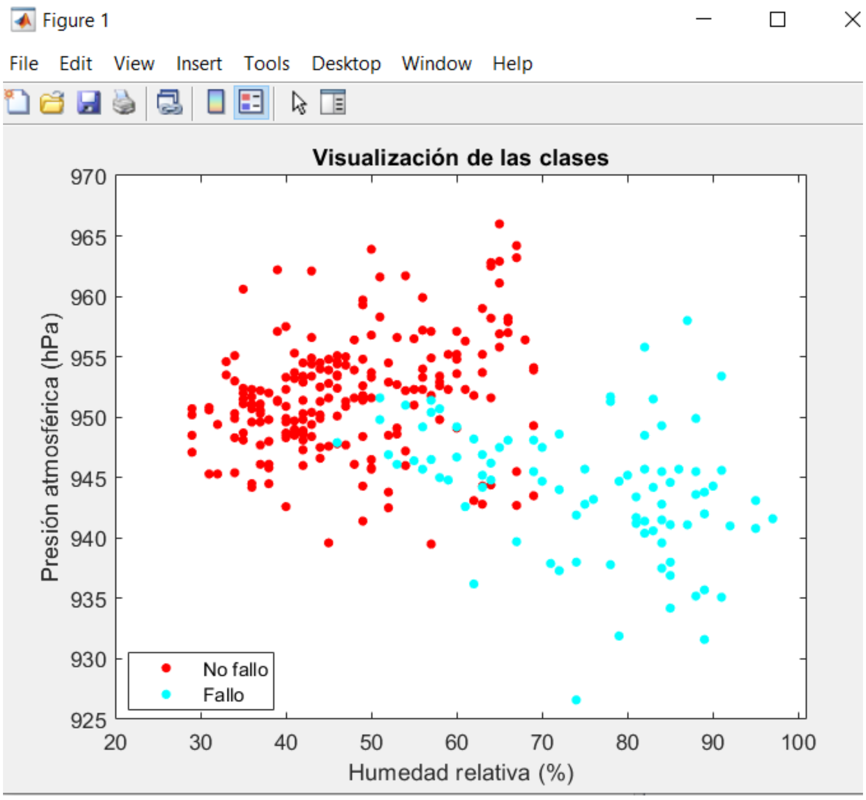

```
plot(x(:,1),x(:,2),'o');
title('Diagramas de dispersión variables de entrada');
xlabel('Humedad relativa(%)');
ylabel('Presión atmosférica (hPa)');
```



Con este código vamos a generar la matriz de contendrá las variables de entrada, “x”, formada por la humedad relativa, “x1”, y la presión atmosférica, “x2”, y el vector de la variable de salida, indicativo del fallo o no de la máquina, “y”. Además genera los histogramas y el gráfico de dispersión.

1.4.3 VISUALIZACIÓN DE LAS CLASE.

```
gscatter(x(:,1),x(:,2),y);
title('Visualización de las clases')
legend('No fallo','Fallo');

ylabel('Presión atmosférica (hPa)');
xlabel('Humedad relativa (%)');
```



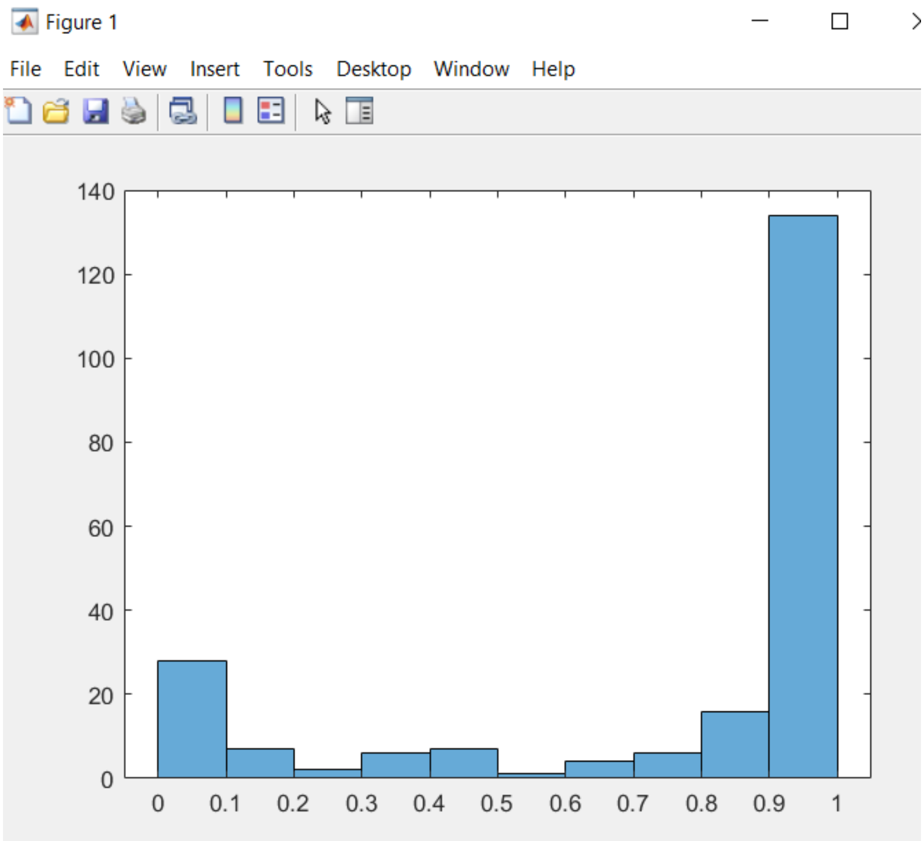
1.4.4 ENTRENAMIENTO DE LA REGRESIÓN LOGÍSTICA.

```
>> P=0.7; % 70 % del conjunto de datos para entrenamiento, luego ejecutar con el
15% para validación y el 10% para test

ans =

    15

>> idx=randperm(m);
>> xtrain=x(idx(1:round(P*m)),:);
>> ytrain=y(idx(1:round(P*m)),:);
>> xtest=x(idx(round(P*m)+1:end),:);
>> ytest=y(idx(round(P*m)+1:end),:);
>> theta=mnrfit(xtrain,ytrain)

theta =

 -183.4221
   -0.1736
    0.2055
```

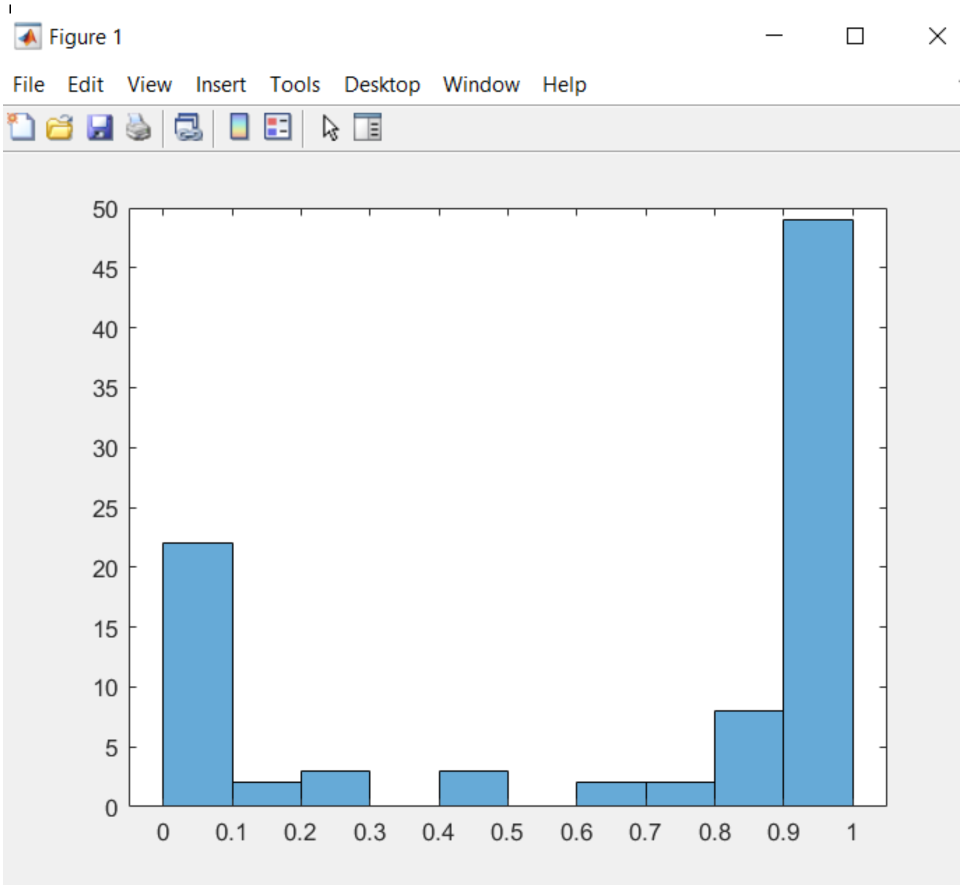
Con este código hemos generado unos resultados de theta que vienen dados por la matriz.

1.4.5 GENERACIÓN DE LA HIPOTESIS PARA EL CONJUNTO DE DATOS DE ENTRENAMIENTO Y PRUEBA.

```
%Entrenamiento
mtrain=length(ytrain);
xtrain2=[ones(mtrain,1) xtrain];
ztrain=xtrain2*theta; % z=theta0*1 + theta1*x1 + theta2*x2
htrain=1.0./(1.0+exp(-ztrain));
histogram(htrain,10);
```



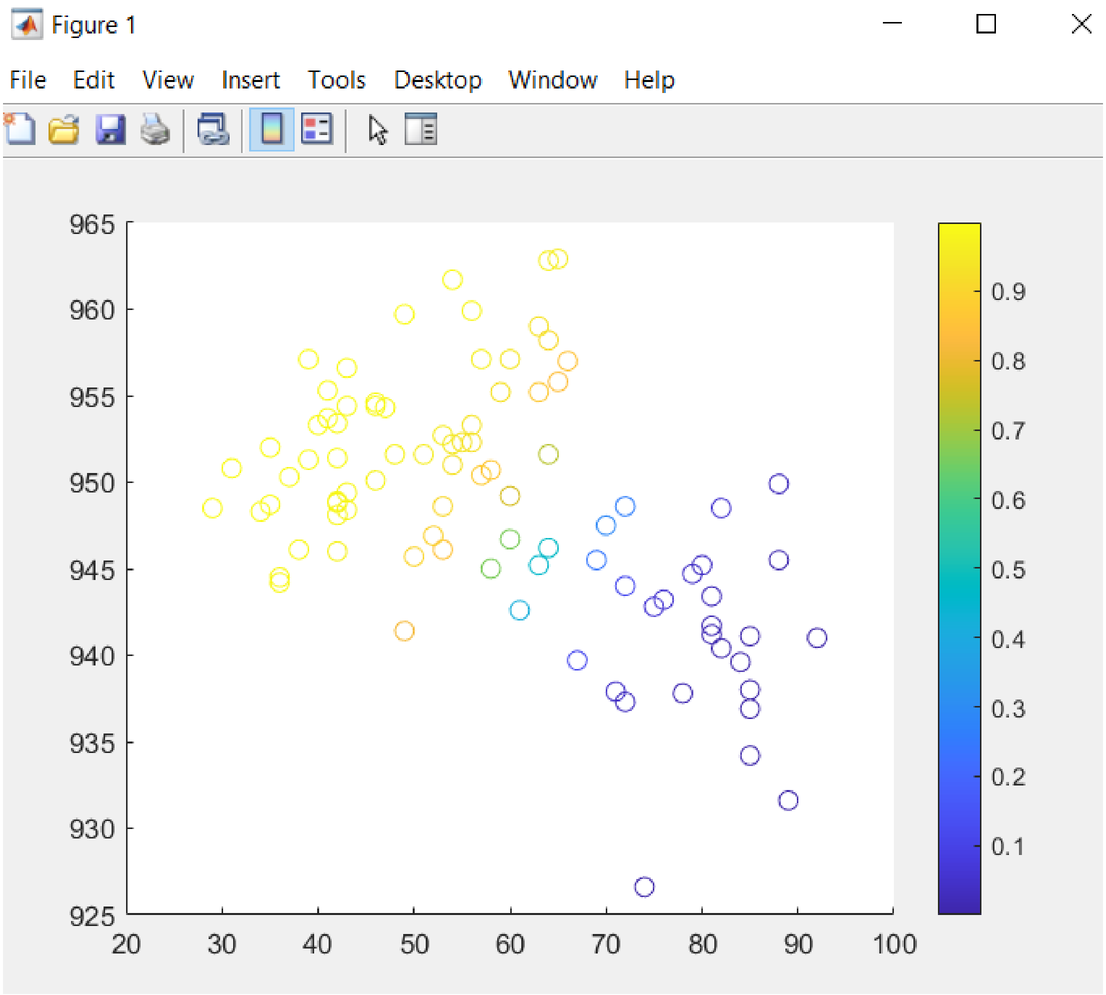
```
%test
mtest=length(ytest);
xtest2=[ones(mtest,1) xtest];
ztest=xtest2*theta; % z=theta0*1 + theta1*x1 + theta2*x2
htest=1.0./(1.0+exp(-ztest));
histogram(htest,10);
```



Con este código generamos los diferentes histogramas de cada hipótesis.

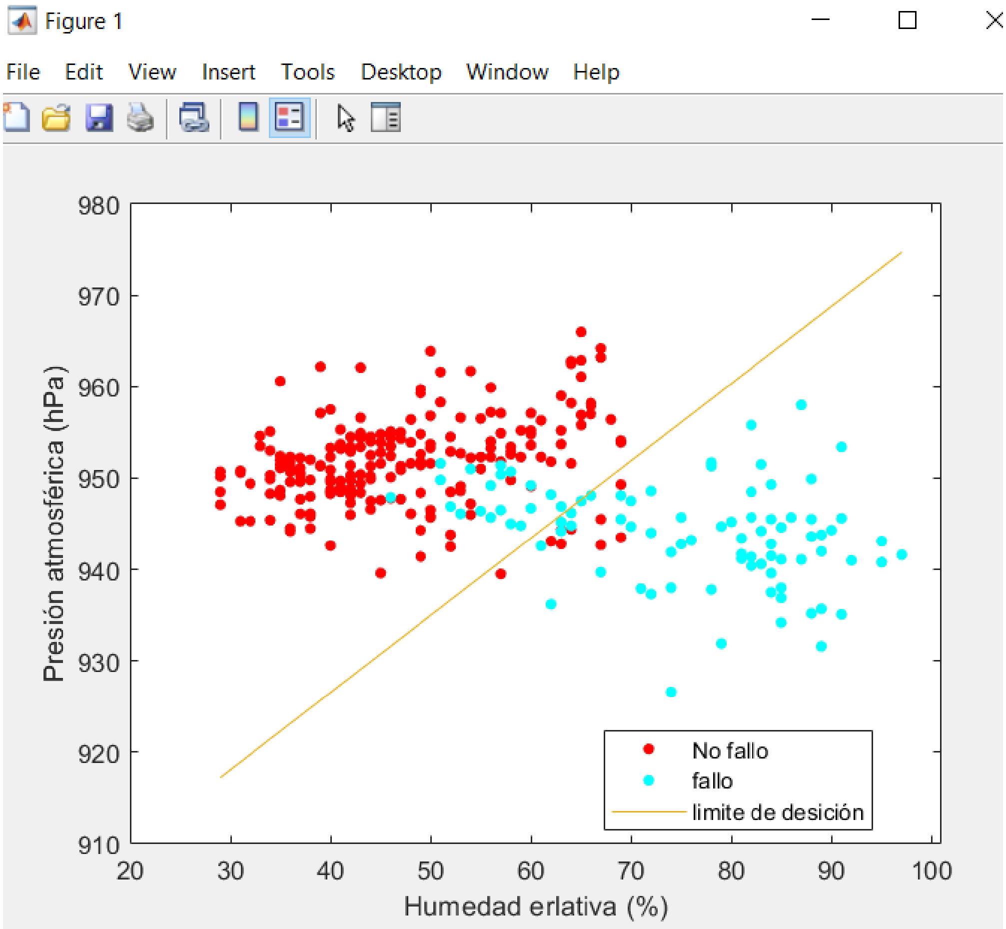
1.4.5 VISUALIZACIÓN DE LA SALIDA DEL MODELO DE CLASIFICACIÓN.

```
scatter(xtest(:,1),xtest(:,2),50,htest);
cb=colorbar();
```



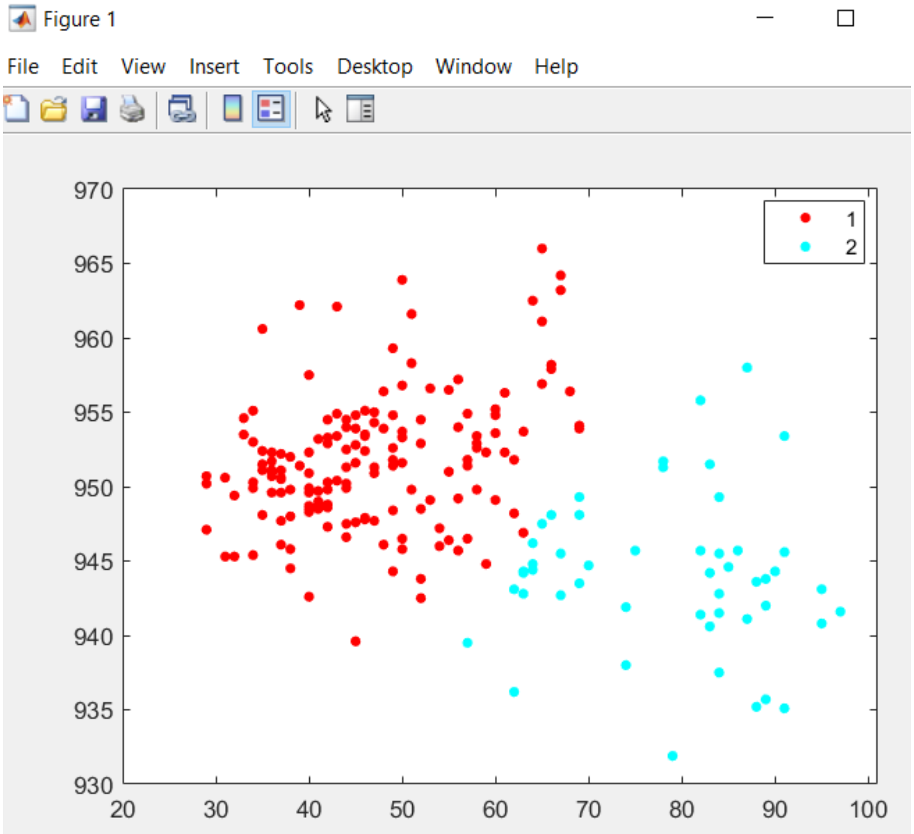
1.4.6 VISUALIZACIÓN DEL LÍMITE DE DECISIÓN.

```
gscatter(x(:,1),x(:,2),y);
legend('No fallo','Fallo');
xlabel('Humedad erlativa (%)');
ylabel('Presión atmosférica (hPa)');
hold on;
plot(x(:,1),-(theta(1)*1 + theta(2)*x(:,1))/theta(3));
legend('No fallo','fallo','limite de desición');
hold off;
```



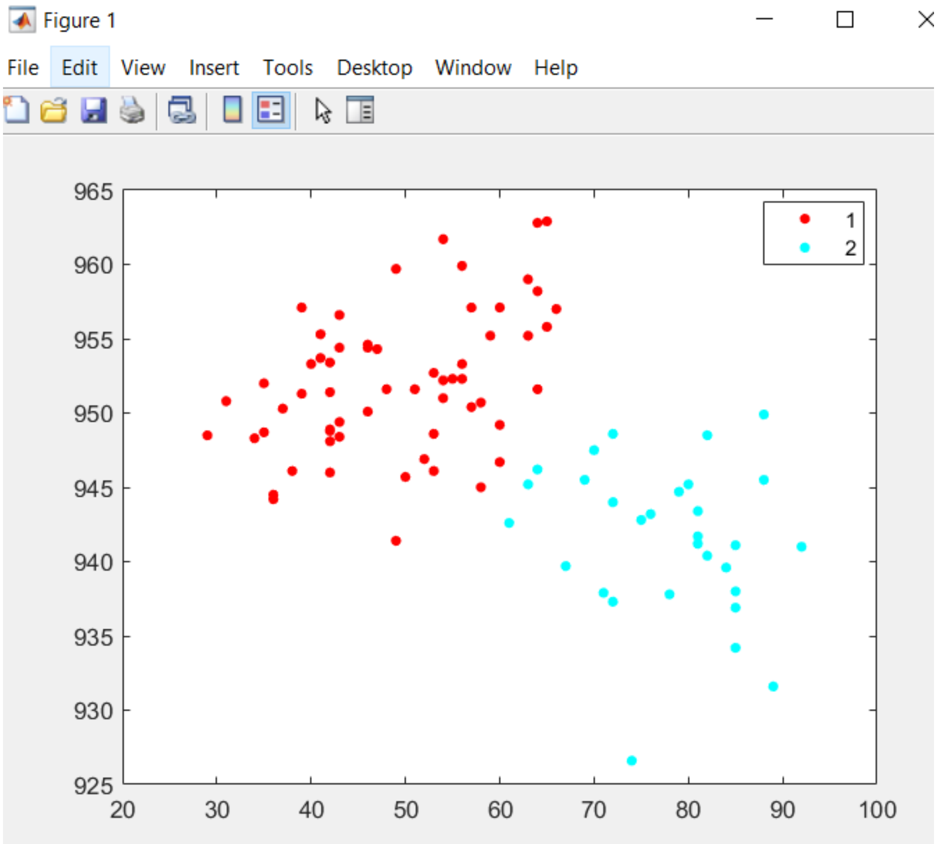
1.4.7 EVALUACIÓN DEL MODELO.

```
%Entrenamiento
ytrainpred=htrain<0.5; % para asegurar más se podía comparar con 0,7 pero no se va a hacer
ytrainpred=ytrainpred+1;
gscatter(xtrain(:,1),xtrain(:,2),ytrainpred);
```




```
>> exactitud_training=mean(double(ytrainpred==ytrain))*100

exactitud_training =

   90.9953
```

```
%test
ytestpred=htest<0.5; % para asegurar más se podía comparar con 0,7 pero no se va a hacer
ytestpred=ytestpred+1;
gscatter(xtest(:,1),xtest(:,2),ytestpred);
```




```
>> exactitud_test=mean(double(ytestpred==ytest))*100

exactitud_test =

   90.1099
```

Precisión y recuperación

```
>> verdad_fallo=sum(double(ytest==2))

verdad_fallo =

    39

>> predice_fallo=sum(double(ytestpred==2))

predice_fallo =

    30

>> verdadero_positivo=sum(double(ytest==2).*double(ytestpred==2))

verdadero_positivo =

    30

>> precision=verdadero_positivo/predice_fallo

precision =

     1

>> recuperacion=verdadero_positivo/verdad_fallo

recuperacion =

    0.7692
```

Como nos debía salir, la exactitud del entrenamiento nos ha salido con un 90,9953% mientras que la exactitud del test nos ha salido 90,1099%.
Como tenemos una precisión de 1, el 100% de los casos serían clasificados como verddaderos positivos. Pero como tenemos una recuperación del 0.7692, tendríamos verdaderos positivos el 76.92% de las veces que se los ha calificado como verdaderos positivos.

1.4.8 EVALUACIÓN DE VARIOS MODELOS.

Los modelos son las thetas, seleccionar el que mejor se comporte

```
>> f1=2*precision*recuperacion/(precision+recuperacion)

f1 =

    0.8696
```

1.5 CUESTIÓN 5.

1.5.1 INTRODUCCIÓN.

Realmente la cuestión 4, era un ejemplo sencillo de aplicación de la regresión logística, para poder afrontar esta nueva cuestión, es decir, se va a predecir si una máquina falla en un tiempo inmediato, en base al estado de cuatro variables que se han controlado durante un largo tiempo y de las cuales depende su correcto funcionamiento, que han dado lugar a una base de datos recogida en el fichero cuestion45_data.xlsx.

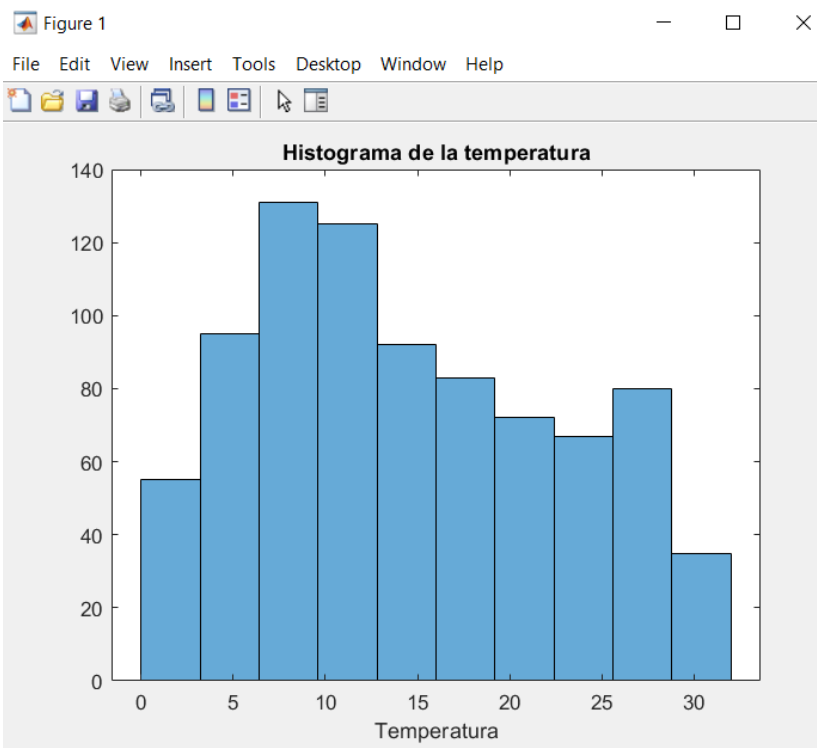
Se pide, en base a lo realizado en la cuestión 4, llevar a cabo esta nueva cuestión, teniéndose en cuenta que algunas representaciones graficas no se podrán llevar a cabo, debido a que no estamos en 2D, es decir, se tienen más de dos variables.

1.5.1 CARGA DE DATOS

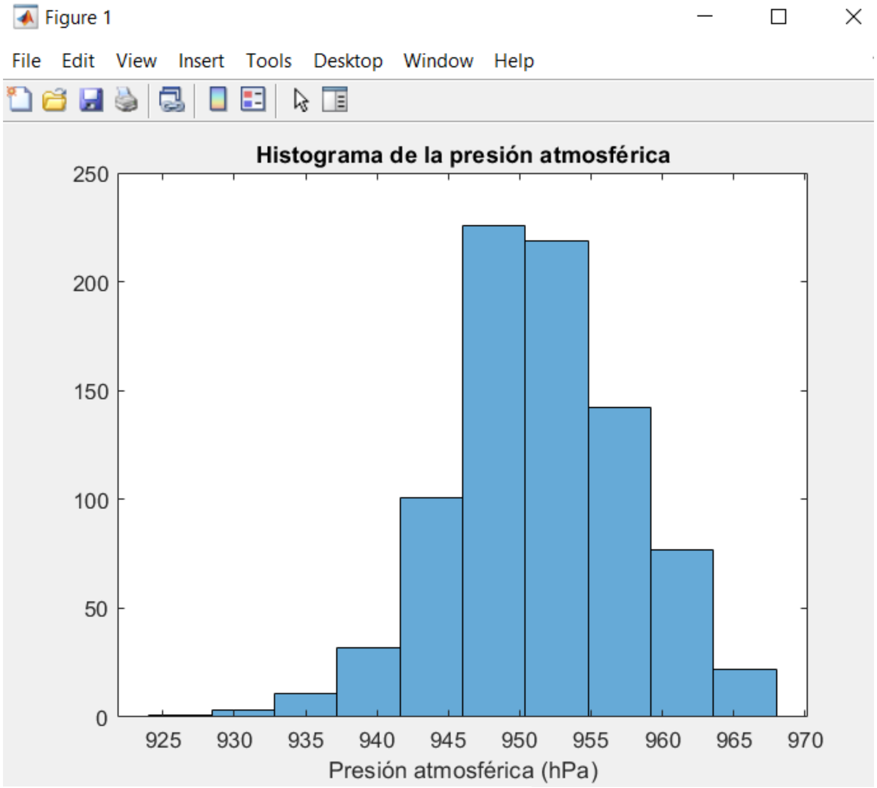
```
datos=readmatrix('cuestion45_data.xlsx');
x=datos(:,1:2);
z=datos(:,3:4);
y=datos(:,5);
y=y+1;
m=length(y);
```

1.5.2 VISUALIZACIÓN DE DATOS.

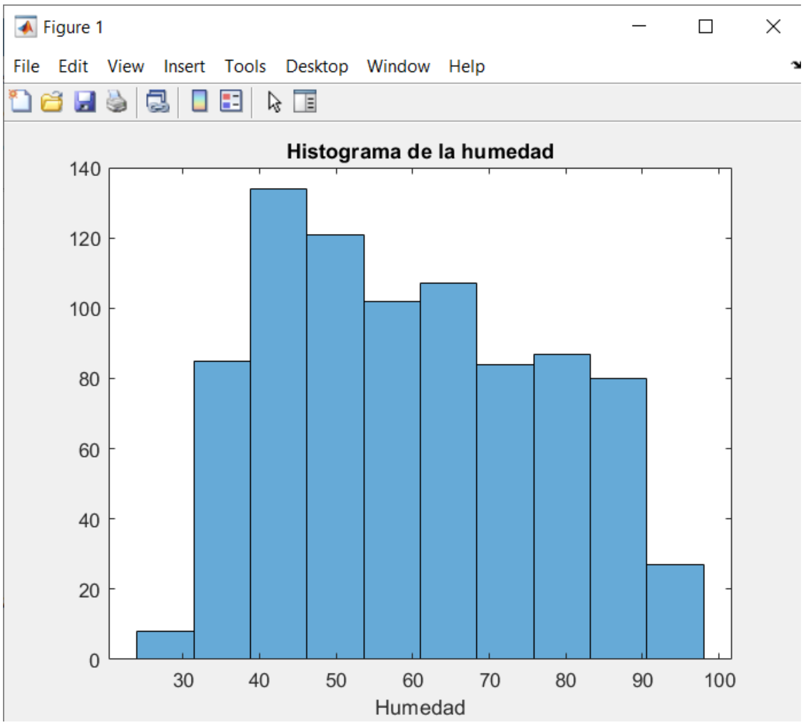
```
histogram(x(:,1),10);
title('Histograma de la temperatura');
xlabel('Temperatura');
```



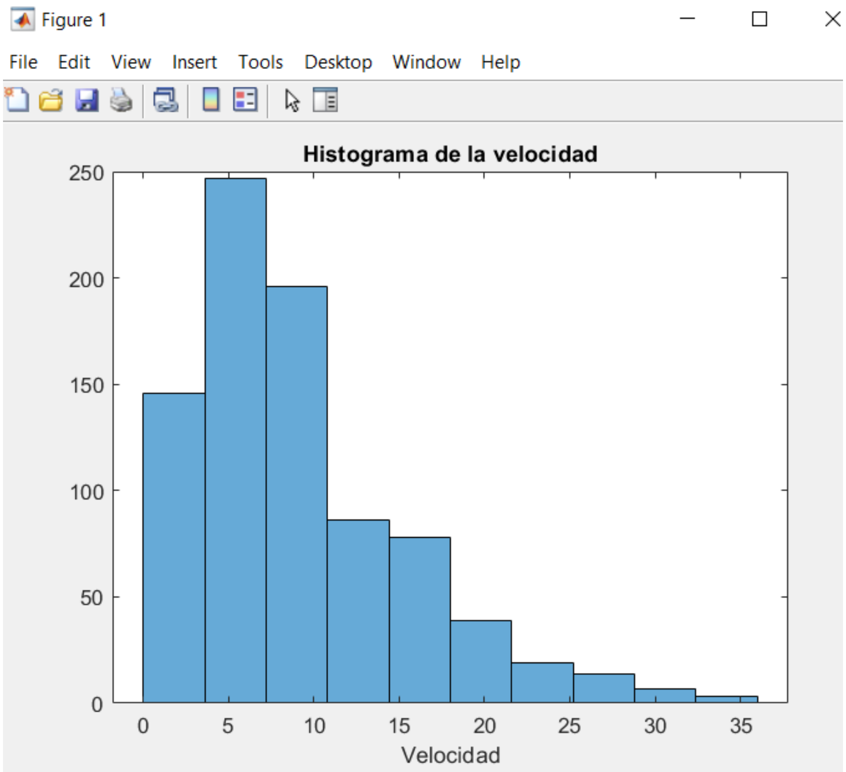
```
histogram(x(:,2),10);
title('Histograma de la presión atmosférica');
xlabel('Presión atmosférica (hPa)');
```



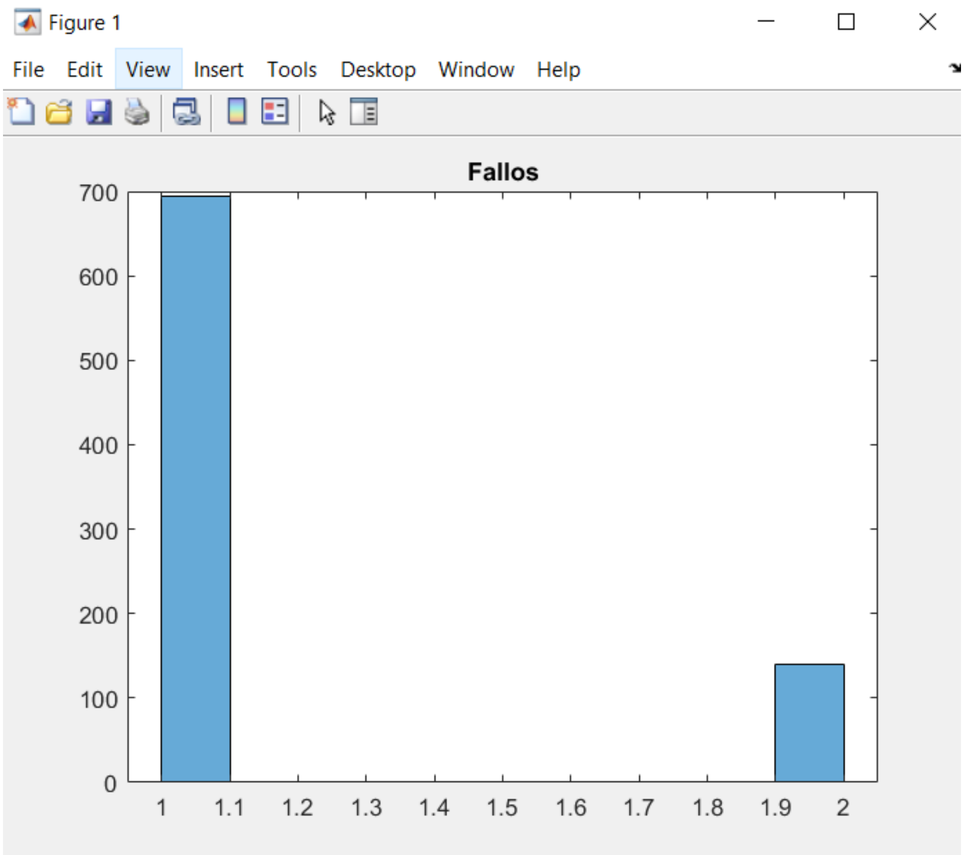

```
histogram(z(:,1),10);
title('Histograma de la humedad');
xlabel('Humedad');
```



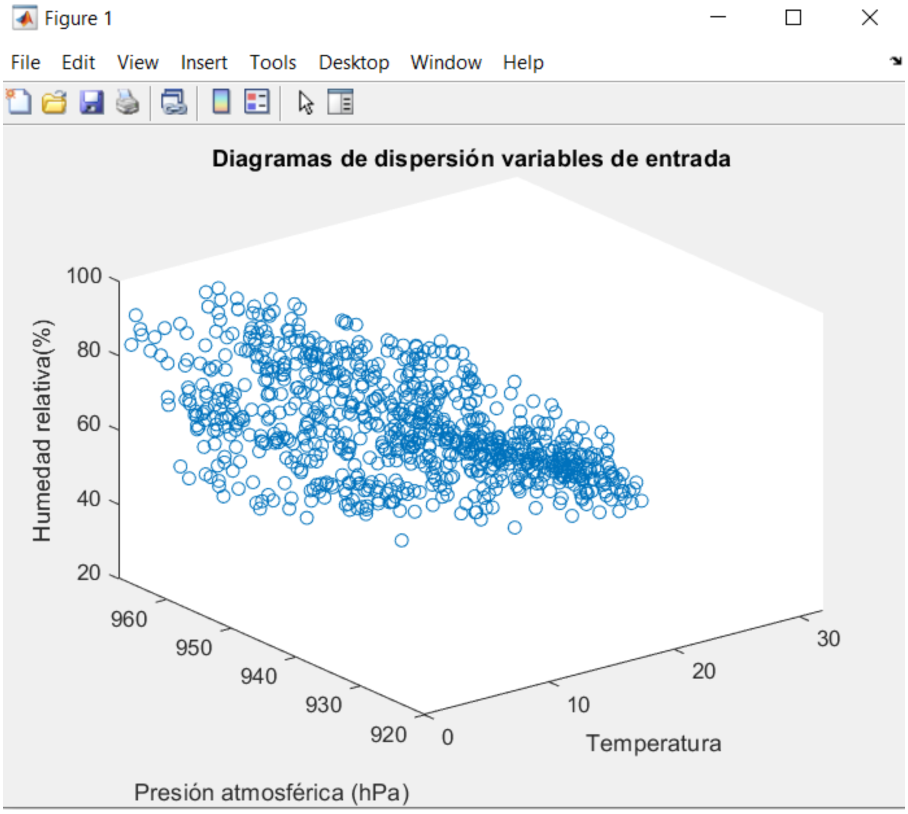
```
histogram(z(:,2),10);
title('Histograma de la velocidad');
xlabel('Velocidad');
```




```
histogram(y(:),10);
title('Fallos');
```

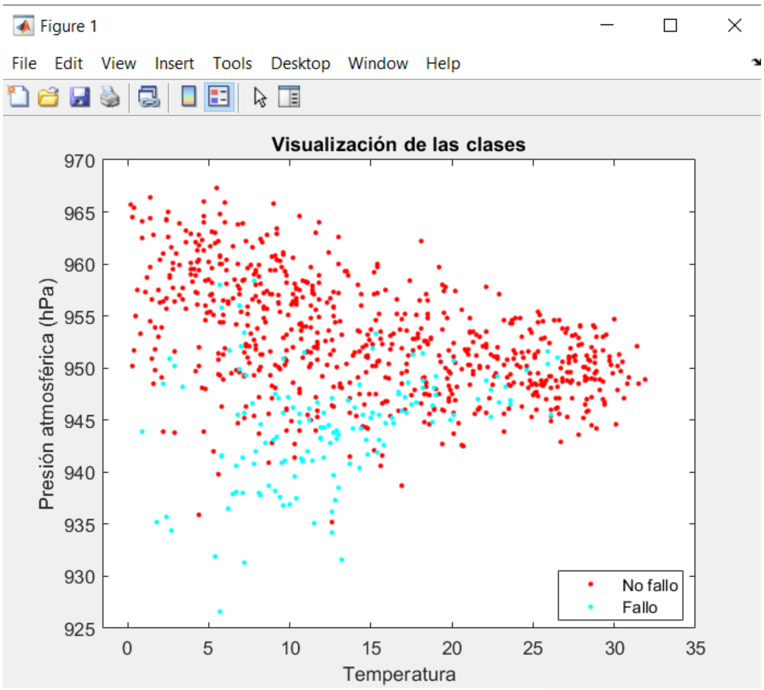


```
plot3(x(:,1),x(:,2),z(:,1),'o');
title('Diagramas de dispersión variables de entrada');
zlabel('Humedad relativa(%)');
ylabel('Presión atmosférica (hPa)');
xlabel('Temperatura');
```

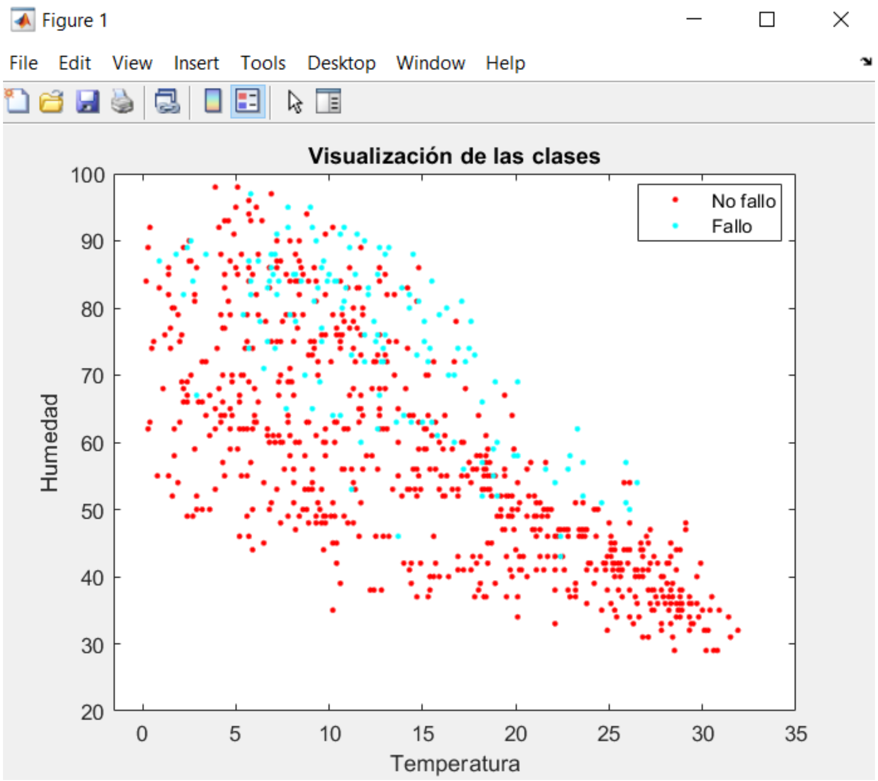


1.5.3 VISUALIZACIÓN DE LAS CLASES.

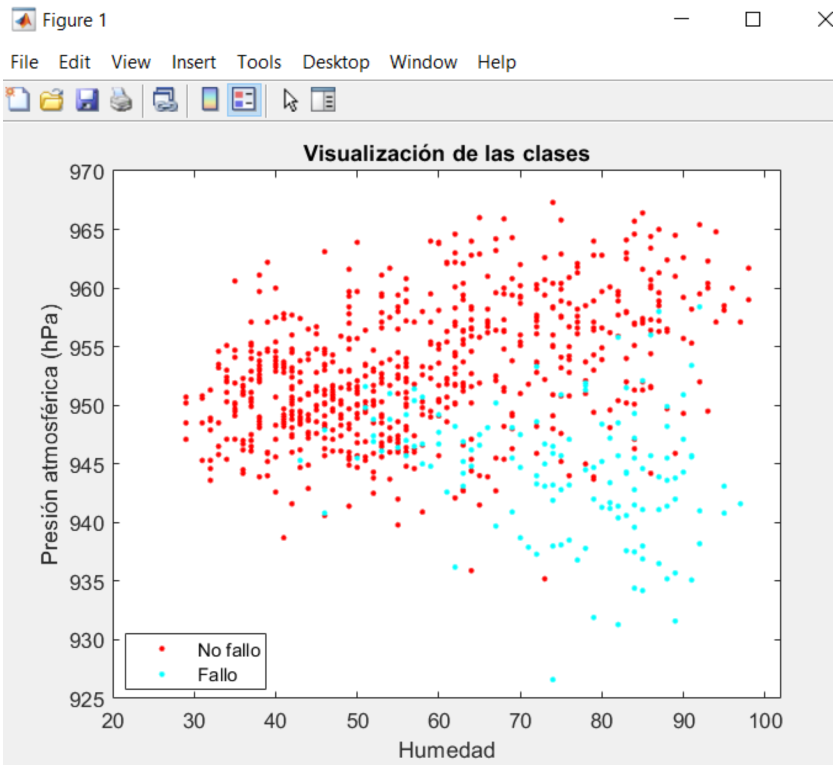
```
gscatter(x(:,1),x(:,2),y);
title('Visualización de las clases');
legend('No fallo','Fallo');
xlabel('Temperatura');
ylabel('Presión atmosférica (hPa)');
```




```
gscatter(x(:,1),z(:,1),y);
title('Visualización de las clases');
legend('No fallo','Fallo');
xlabel('Temperatura');
ylabel('Humedad');
```



```
gscatter(z(:,1),x(:,2),y);
title('Visualización de las clases')
legend('No fallo','Fallo');
xlabel('Humedad');
ylabel('Presión atmosférica (hPa)');
```

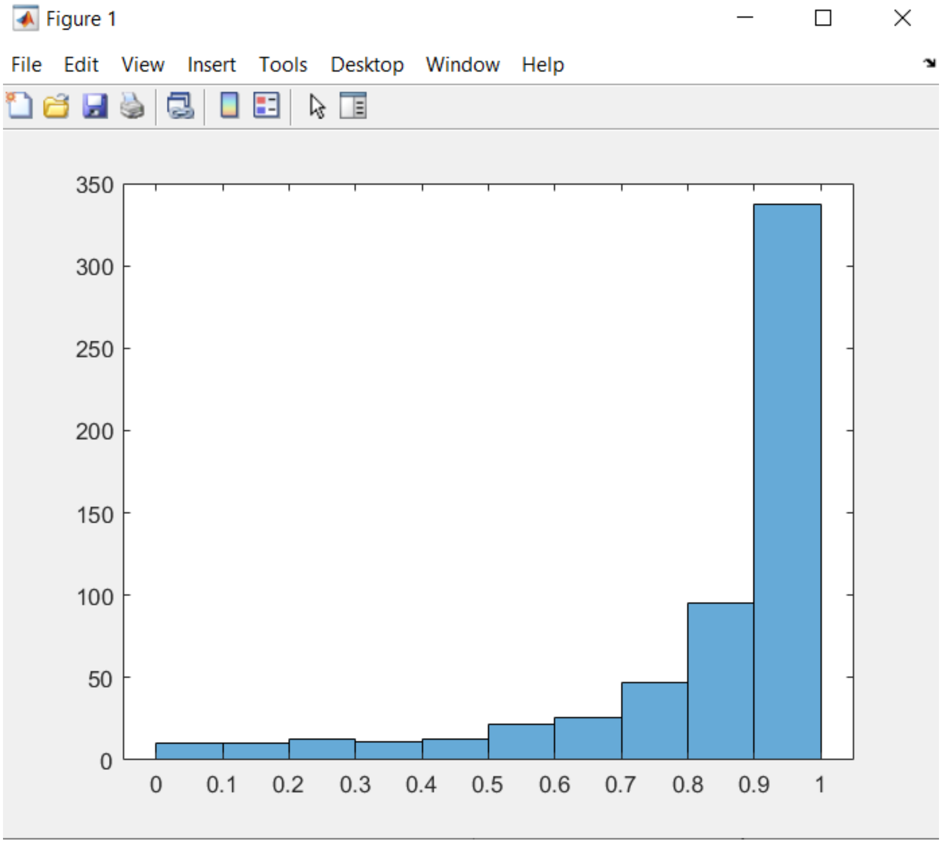


1.5.4 ENTRENAMIENTO DE LA REGRESIÓN LOGÍSTICA.

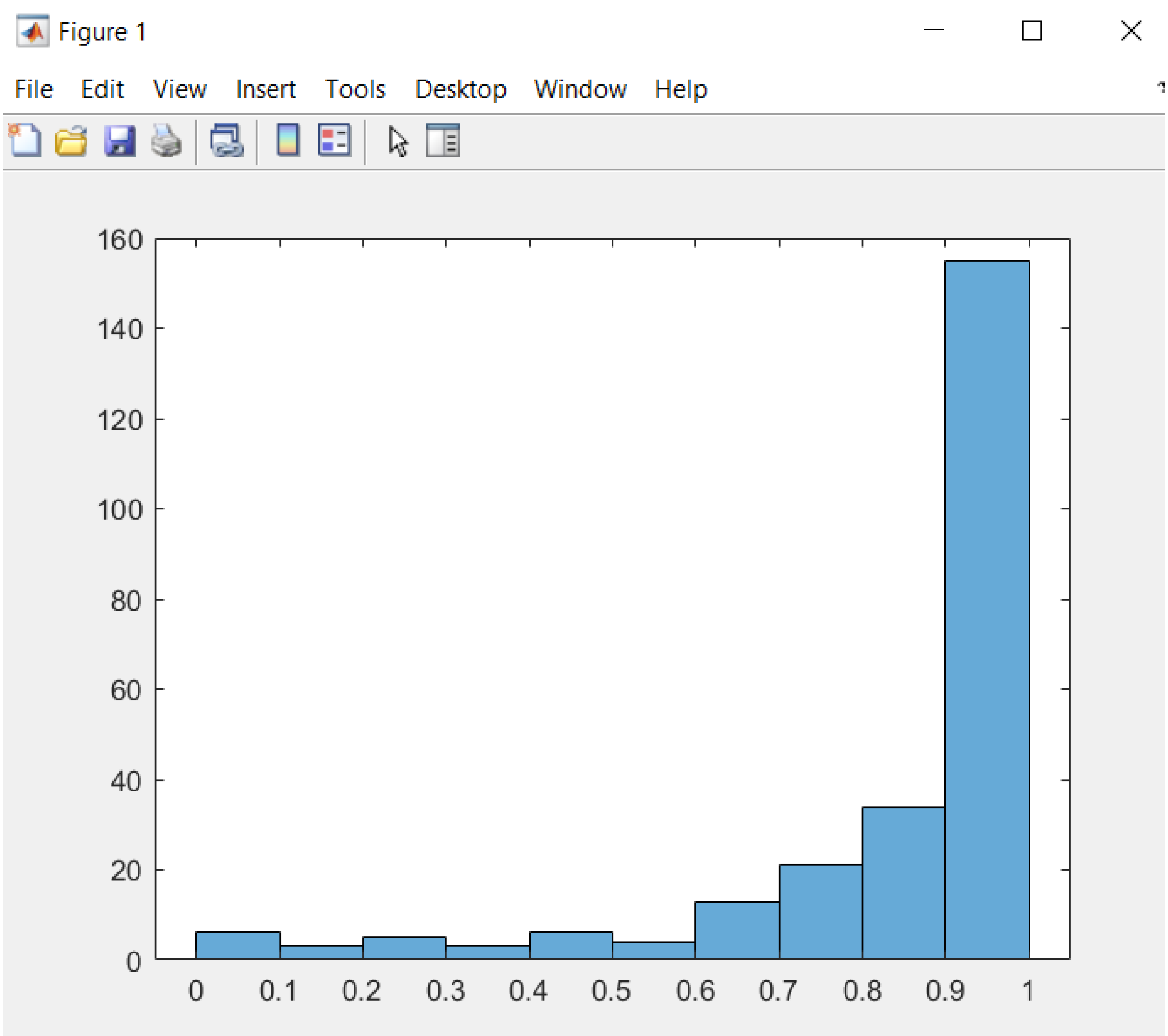
```
>> P=0.7; % 70 % del conjunto de datos para entrenamiento, luego ejecutar con el 15% para validación y el 10% para test
>> idx=randperm(m);
>> xtrain=x(idx(1:round(P*m)),:);
>> ytrain=y(idx(1:round(P*m)),:);
>> xtest=x(idx(round(P*m)+1:end),:);
>> ytest=y(idx(round(P*m)+1:end),:);
>> theta=mnrfit(xtrain,ytrain)

theta =

 -281.8786
    0.0720
    0.2977
```

1.5.5 GENERACIÓN DE LA HIPOTESIS PARA EL CONJUNTO DE DATOS DE ENTRENAMIENTO Y PRUEBA.

```
%Entrenamiento
mtrain=length(ytrain);
xtrain2=[ones(mtrain,1) xtrain];
ztrain=xtrain2*theta; % z=theta0*1 + theta1*x1 + theta2*x2
htrain=1.0./(1.0+exp(-ztrain));
histogram(htrain,10);
```



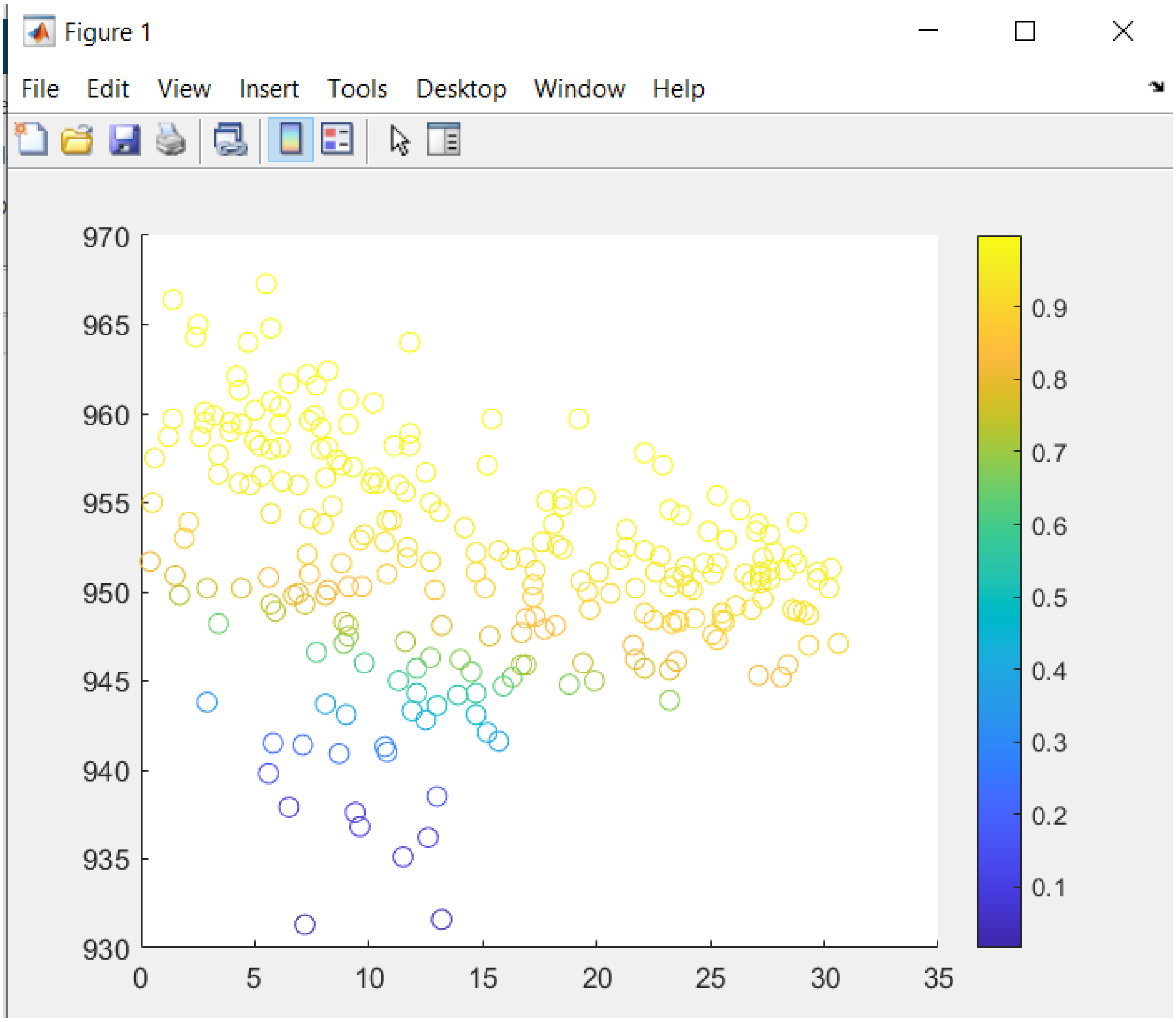
```
%test
mtest=length(ytest);
xtest2=[ones(mtest,1) xtest];
ztest=xtest2*theta; % z=theta0*1 + theta1*x1 + theta2*x2
htest=1.0./(1.0+exp(-ztest));
histogram(htest,10);
```



Con este código sacamos los diferentes histogramas de cada hipótesis.

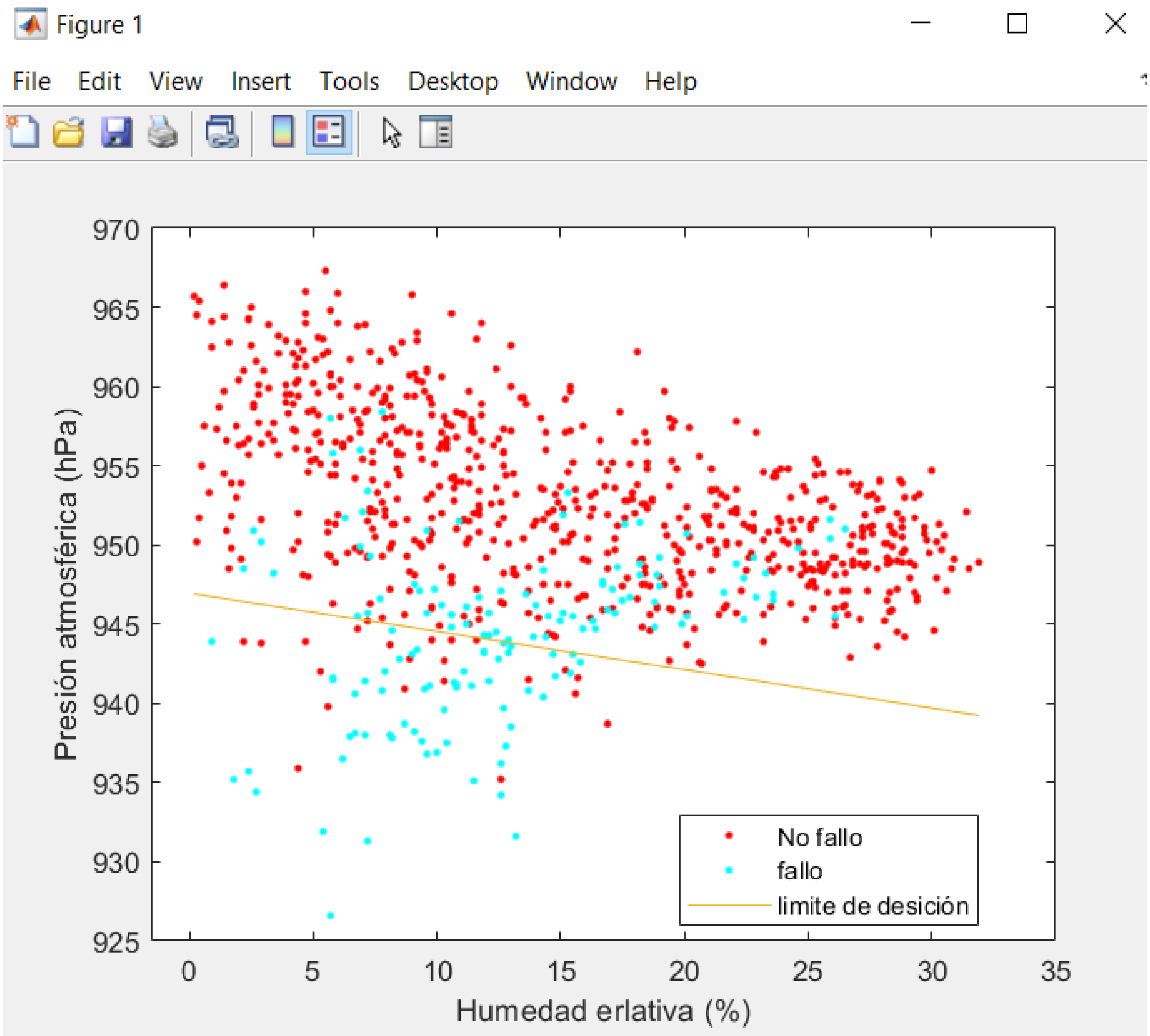
1.5.6 VISUALIZACIÓN DE LA SALIDA DEL MODELO DE CLASIFICACIÓN.

```
scatter(xtest(:,1),xtest(:,2),50,htest);
cb=colorbar();
```



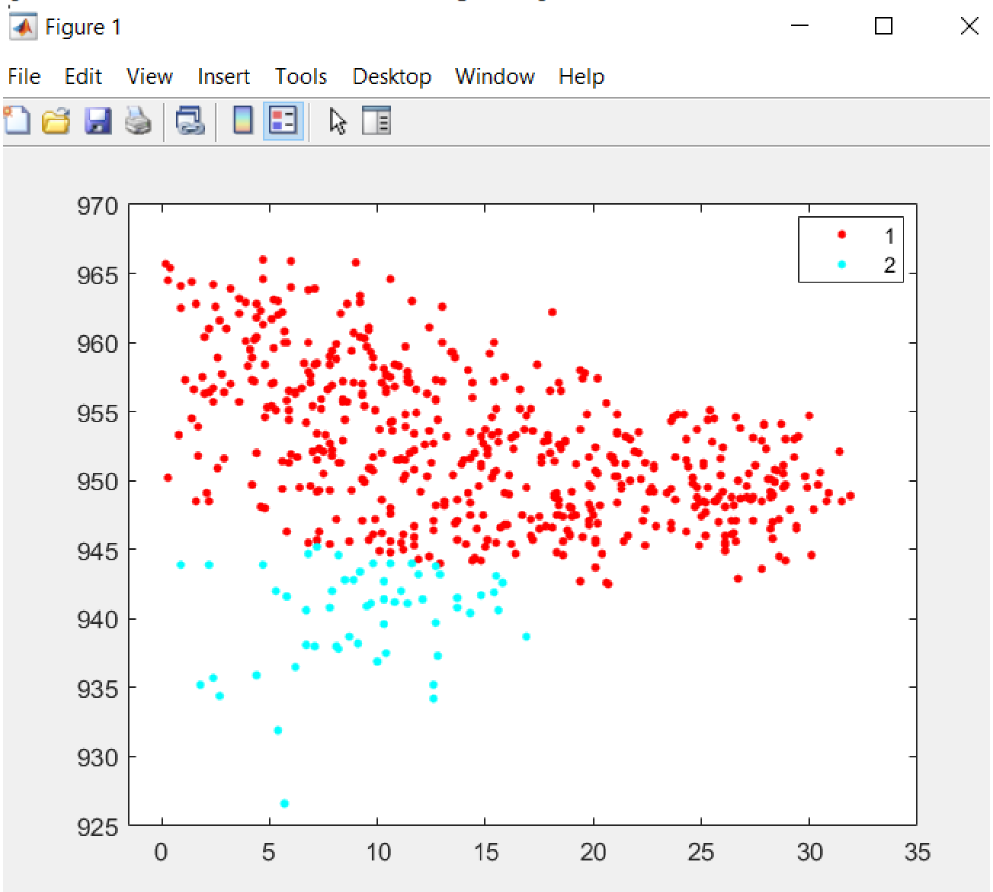
1.5.7 VISUALIZACIÓN DEL LÍMITE DE DECISIÓN.

```
gscatter(x(:,1),x(:,2),y);
legend('No fallo','Fallo');
xlabel('Humedad erlativa (%)');
ylabel('Presión atmosférica (hPa)');
hold on;
plot(x(:,1),-(theta(1)*1 + theta(2)*x(:,1))/theta(3));
legend('No fallo','fallo','limite de desición');
hold off;
```



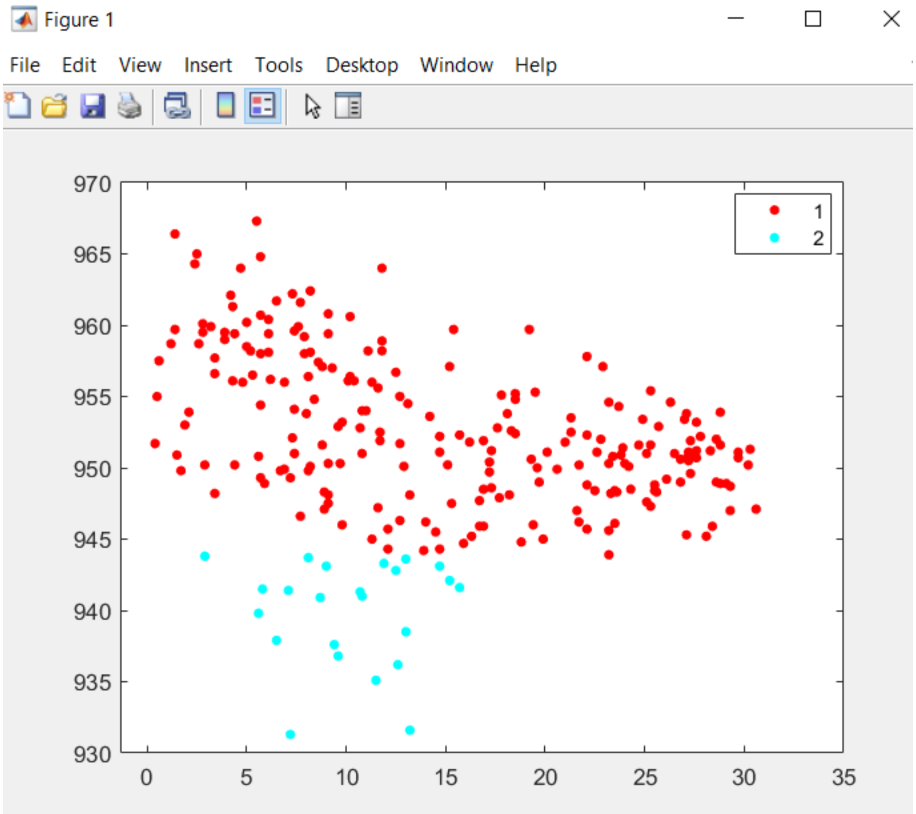
1.5.8 EVALUACIÓN DEL MODELO.

```
%Entrenamiento
ytrainpred=htrain<0.5; % para asegurar más se podía comparar con 0,7 pero no se va a hacer
ytrainpred=ytrainpred+1;
gscatter(xtrain(:,1),xtrain(:,2),ytrainpred);
```




```
>> exactitud_training=mean(double(ytrainpred==ytrain))*100

exactitud_training =

   87.3504
```

```
%test
ytestpred=htest<0.5; % para asegurar más se podía comparar con 0,7 pero no se va a hacer
ytestpred=ytestpred+1;
gscatter(xtest(:,1),xtest(:,2),ytestpred);
```




```
>> exactitud_test=mean(double(ytestpred==ytest))*100

exactitud_test =

    88
```

Precisión y recuperación

```
>> verdad_fallo=sum(double(ytest==2))

verdad_fallo =

    41

>> predice_fallo=sum(double(ytestpred==2))

predice_fallo =

    23

>> verdadero_positivo=sum(double(ytest==2).*double(ytestpred==2))

verdadero_positivo =

    17

>> precision=verdadero_positivo/predice_fallo

precision =

    0.7391

>> recuperacion=verdadero_positivo/verdad_fallo

recuperacion =

    0.4146
```

1.5.9 EVALUACIÓN DE VARIOS MODELOS.

```
>> f1=2*precision*recuperacion/(precision+recuperacion)

f1 =

    0.5312
```

1.6 CUESTIÓN 6.

1.6.1 INTRODUCCIÓN.

En el fichero cuestión46_data.xlsx, se encuentran tres tipos de posibles fallos, que se pueden dar en una máquina, agrupados en tres clases (1, 2 y 3), en función del estado de cuatro variables (x1, x2, x3 y x4), es decir, se trata de un problema de clasificación multiclase o regresión logística multinomial.

En base al algoritmo denominado clasificación "uno contra todos", y teniendo como apoyo lo estudiado en las cuestiones 4 y 5, desarrollar el procedimiento para obtener el algoritmo que sea capaz de clasificar futuros fallos en dichas tres clases.

1.6.1 CUESTIÓN.

```
datos=readmatrix('cuestion46_data.xlsx');
X=datos(:,1:4);
y=datos(:,5);
m=length(y);
```

Creación del conjunto de entrenamiento 70% y prueba 30%

```
P=0.7; % 70 % del conjunto de datos para entrenamiento
idx=randperm(m);
Xtrain=X(idx(1:round(P*m)),:);
ytrain=y(idx(1:round(P*m)),:);
Xtest=X(idx(round(P*m)+1:end),:);
ytest=y(idx(round(P*m)+1:end),:);
```

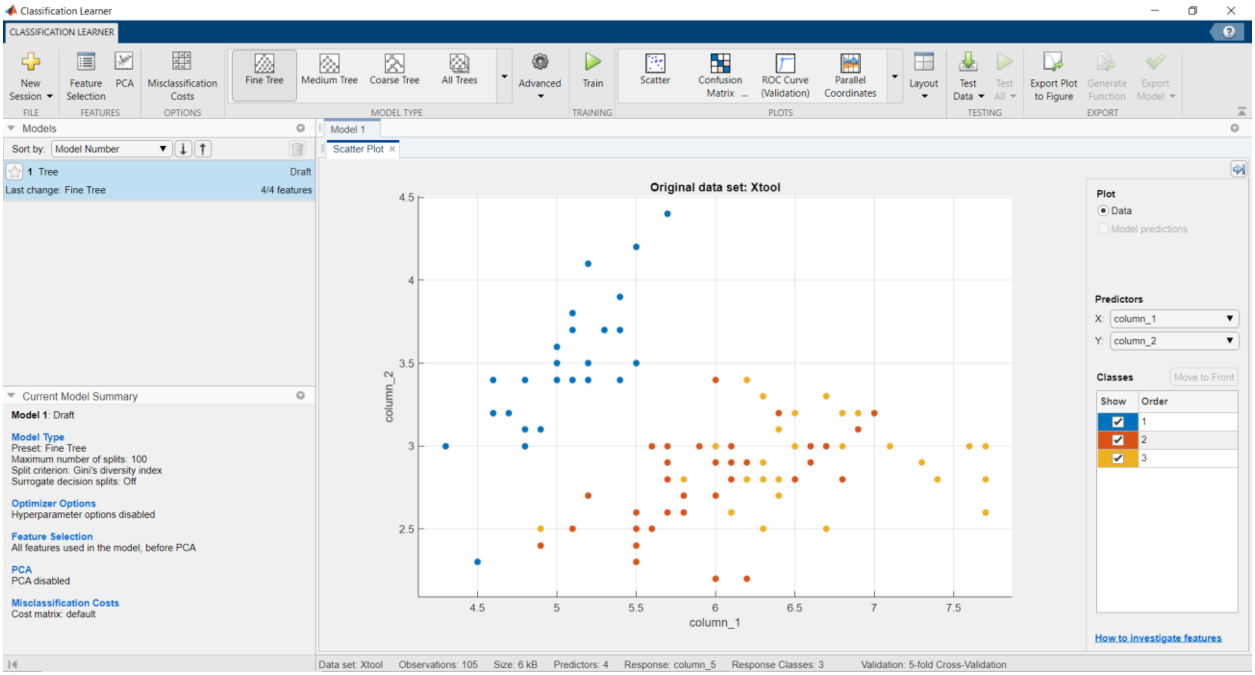
Entrenamiento de la regresión logistica uno contra todos

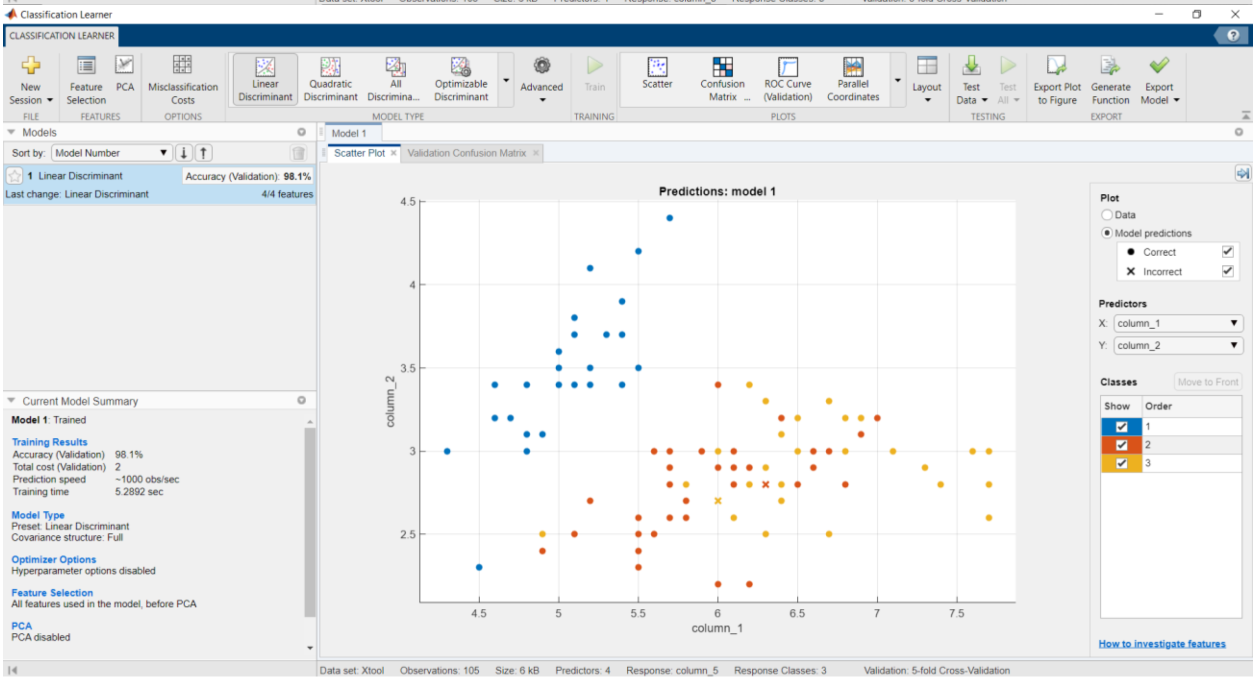
```
theta=mnrfit(Xtrain,ytrain);
mtest=length(ytest);
Xtest2=[ones(mtest,1) Xtest];
predecir(:,1)=exp(Xtest2*theta(:,1))./(1+exp(Xtest2*theta(:,1)+exp(Xtest2*theta(:,2))));
predecir(:,2)=exp(Xtest2*theta(:,2))./(1+exp(Xtest2*theta(:,2)+exp(Xtest2*theta(:,2))));
predecir(:,3)=1./(1+exp(Xtest2*theta(:,1)+exp(Xtest2*theta(:,2))));
[Valor,ytestpred]=max(predecir,[],2);
```

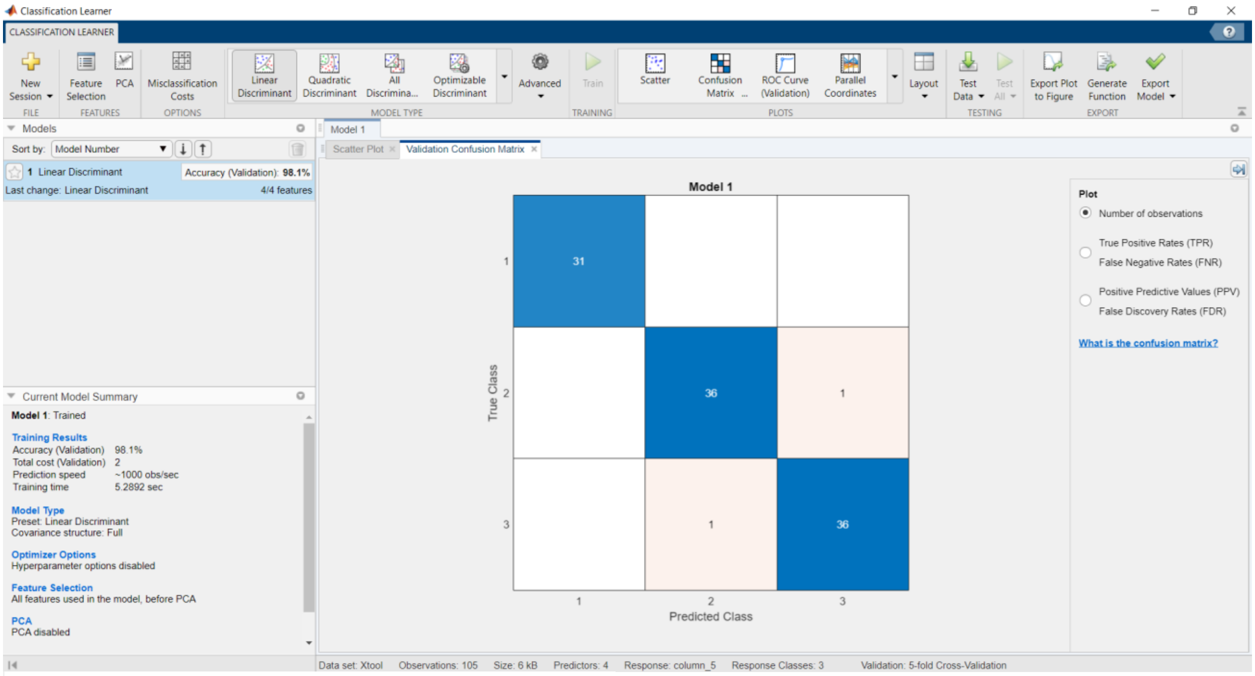
Precisión para el conjunto de prueba

```
>> precision_test=mean(double(ytestpred==ytest))*100

precision_test =

   71.1111
```

APP de Matlab








```
>> Xtool=[Xtrain,ytrain];
>> yfit = trainedModel.predictFcn(Xtest)

yfit =

     3
     2
     2
     1
     2
     1
     1
     3
     2
     1
     3
     1
     3
     1
     2
     1
     3
     1
     3
     2
     3
     1
     3
     3
     3
     2
```

```
1
3
3
1
1
2
2
1
2
1
1
1
2
1
2
3
1
1
3
>> precision_test=mean(double(yfit==ytest))*100

precision_test =

   97.7778
```