UNIVERSIDADE DE AVEIRO
DEPARTAMENTO DE ELECTRÓNICA TELECOMUNICAÇÕES E INFORMÀTICA

Machine Learning (2016/17) – Lab work 1

Objectives: Working with data in Octave/MATLAB. Polynomial approximation of data.

1. Load and plot the Data

The file *data.txt* contains 2D coordinates of real valued points. Create a main script to load the data into variables *x* (the first column) and *y* (the second column). How many points are collected in the file? Write a function *plotData(x,y)* to create
1) One figure with the scatter plot of data with red crosses.
2) Second figure with the histograms of *x* and *y*.
3) Add labels, titles, legends to understand better the plots.
After *plotData(x,y)* is executed in the main script it is expected to see figures similar to Fig. 1 and Fig. 2. Compute the percentage of points with negative coordinates x and y.

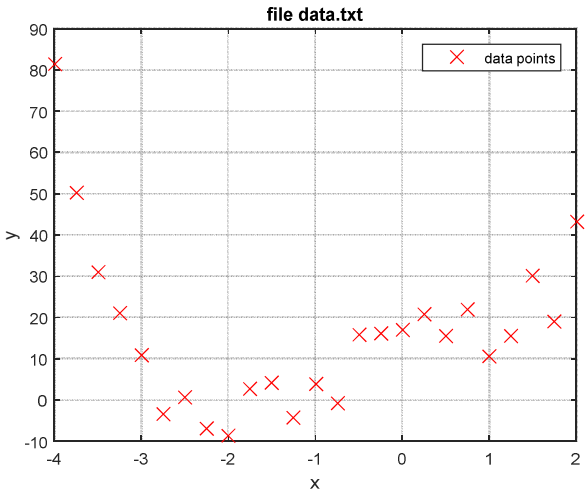

Fig. 1

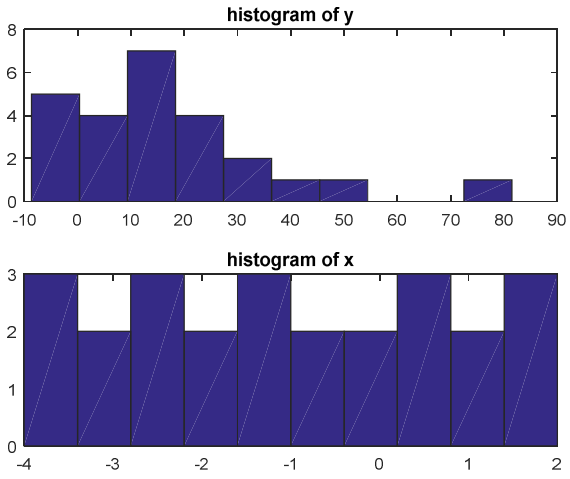

Fig. 2

2. Polynomial approximation

Find a polynomial function (model, hypothesis) that approximates the points (*x, y*):

$$y \approx f(x) = \theta_n x^n + \theta_{n-1} x^{n-1} + \theta_{n-2} x^{n-2} ..... + \theta_1 x + \theta_0$$

Choose the order of the polynomial n such that the mean squared error (MSE) $e = \sum_{i=1}^{m} \left(y^{(i)} - f(x^{(i)}) \right)^2 < 1$, m is the number of data points.

Suggestion: Use a *while* loop and save the errors *e(n)* for each polynomial order n in a vector. Use functions of Matlab/Octave polyfit and polyval. Plot the error vector.

3. Data normalization
Normalize the values of *x* and *y* such that abs(x)<1 and abs(y)<1.
Call function *plotData(xnorm, ynorm)* with the normalized data. Repeat ex. 2.

4. Original data versus normalized data
Compute the mean squared errors between y and the polynomial approximation of order 1 to 30 (suggestion: use a *for* loop) for the original and normalized data and plot the errors as shown in Fig. 3.

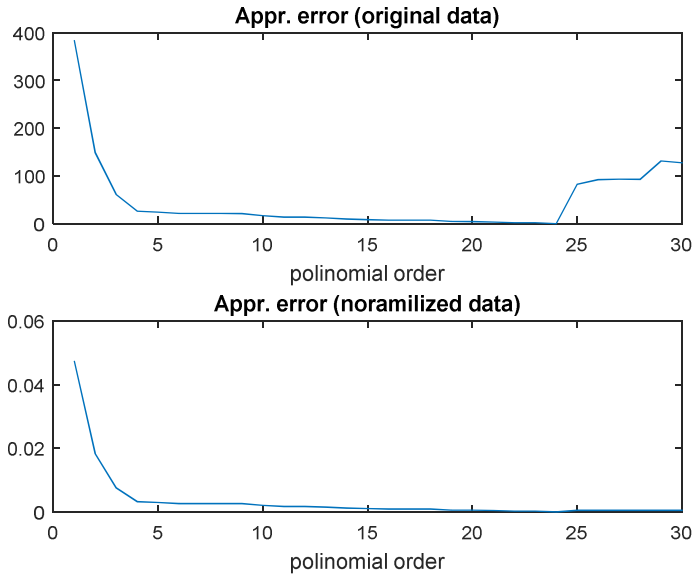

Fig.3

5. Create a matrix S with 3 columns according to the following specifications: the first column is equal to x, the second column contains the elements of x in inverse order and the third column is the mean of the first two columns.

6. Generate a matrix M with 5 rows and 4 columns with random elements uniformly distributed on the interval (-1, 1), use the build-in function *rand* of Octave/Matlab. Generate a matrix N with 4 rows and 3 columns with normally distributed random elements having -2 mean and 0.5 variance, use the build-in function *randn* of Octave/Matlab. Compute the product of the matrices P=M*N and the percentage of positive elements of P.