FGV EMAp
ESCOLA DE
MATEMÁTICA
APLICADA

Machine Learning 1

Assessed Coursework 1

This assignment must be submitted by January 30th, 2023 at 10:00 am.
Late submissions will penalized of 10% per hour of delay.

This coursework is assessed and mandatory
and is worth 30% of your total final grade for this course.

Instructions

Identifier

Please choose a random number of 6 digits. Make sure that you keep a copy of that number as it will be used to provide the feedback (please avoid trivial numbers, such as 000000 or 123456. Also please avoid numbers starting with zero).

Submission

Compress the files of your submission into a unique zip file and rename it with your random digit number (so the zip file name becomes something like 723923.zip). Then email *your zip file as an attachment at alberto.paccanaro@fgv.br with the subject "URGENT – MSC COURSE – COURSEWORK 1 SUBMISSION".*

All the work you submit should be solely your own work.
Coursework submissions will be checked for this.

Note: the first 4 exercises in this coursework can easily be implemented without loops…

EXERCISE 1 (2 marks)

Write a function called *evenodd* that will take in input a natural number *n* and a value *e* in the set {1,2} and then:

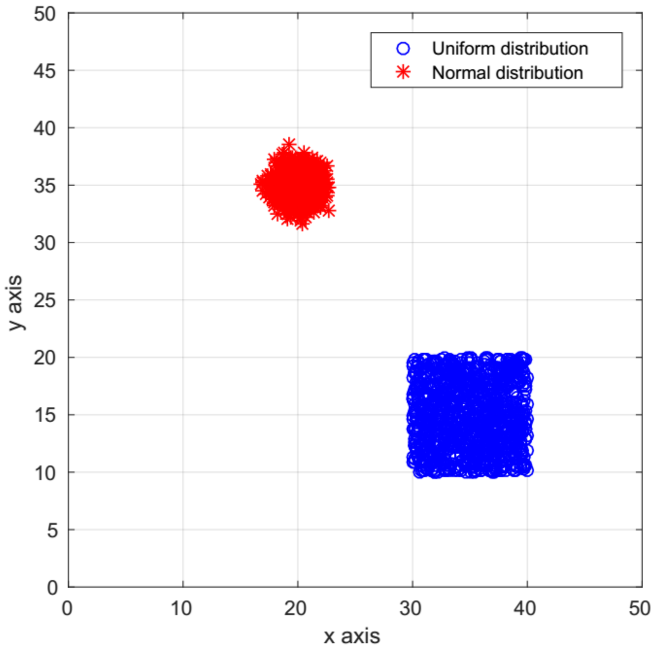
- Create a vector *v* of length *n* of random integers in the range [0 .. 30]
- If *e* is 1, return only the elements of *v* which have odd values and are placed at odd positions in *v*.
- If *e* is 2, return only the elements of *v* which have even values and are placed at even positions in *v*.

EXERCISE 2 (4 marks)

Create a script that will:

- Generate 1000 points, in 2 dimensions, which are uniformly distributed in the range: xmin=30, xmax=40, ymin=10, ymin=20.
- Plot the points, as blue circles, in a figure whose axis are set in the range xmin=0, xmax=50, ymin=0, ymin=50
- Add to the same figure 1000 points, in 2 dimensions, which are normally distributed with a mean value of (20, 35) and unit variance. These points should be denoted by red stars.
- Add the legend and the axis labels (as shown in the figure below).

Your figure should look something like the one given below.



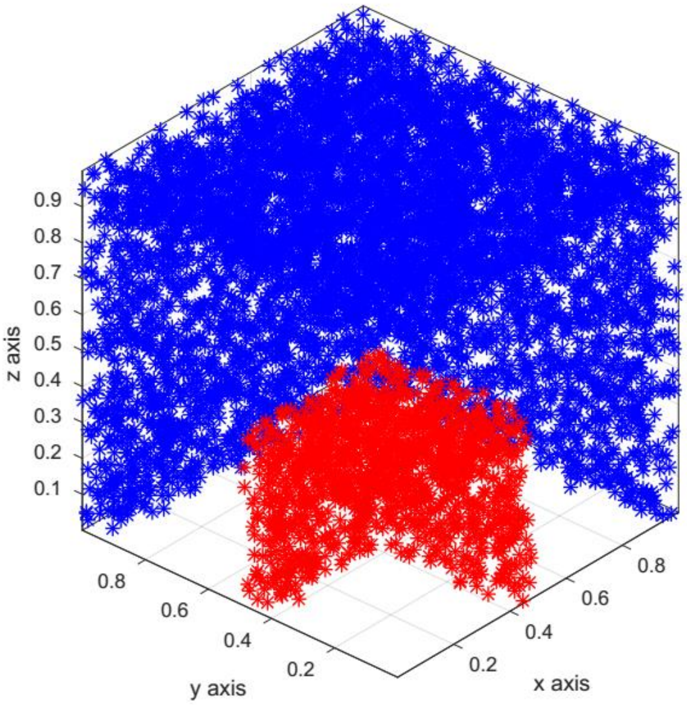
EXERCISE 3 (6 marks)

Create a script that will:

- Generate 20000 points, in 3 dimensions, uniformly distributed in the interval (0, 1).
- Extract those points for which at least one of the 3 dimensions is bigger than 0.9 and then plot them as blue stars in 3 dimensions.
- Extract those points for which both conditions are met:
    (1) at least one of the 3 dimensions is bigger than 0.4
    (2) all the dimensions are smaller than 0.5

  and then plot them as red stars in 3 dimensions.
- Add the axis labels (as shown in the figure below).

Your figure should look something like the one given below.



EXERCISE 4 (Value: 8 marks)

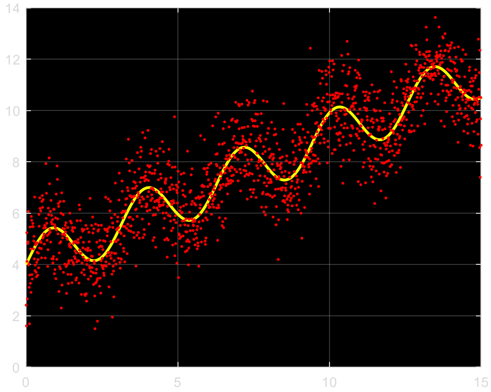
Assume that we have a process following the equation:

y = sin(2x) + 1/2 x + 4 with x in [0, 15]

We can measure y at intervals of 0.01, but these values are corrupted by a Gaussian noise with zero mean and unit variance. Write a script that implements the following points 1-7:

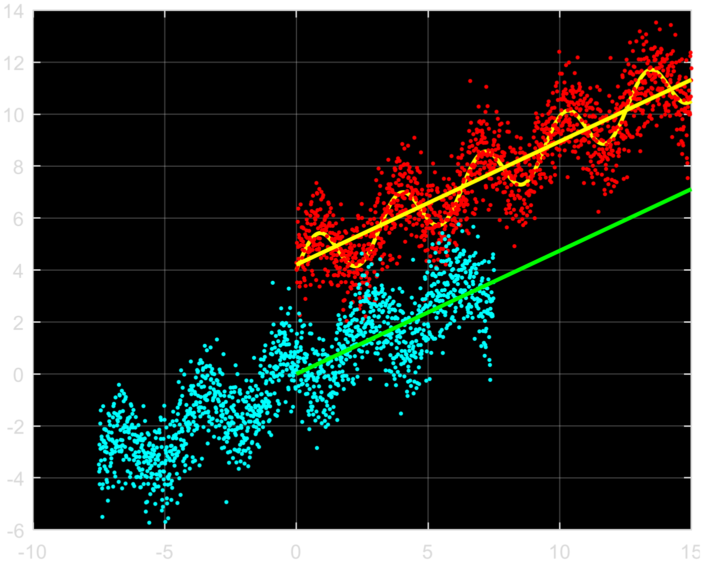
1) Create the dataset
2) Plot the dataset as red dots
3) Plot the above equation in yellow

(after point 3, your plot should look something like the figure below)



4) mean centre the data and plot it in cyan (add it to your existing drawing)
5) assuming that *y = x * w + noise* learn w by least squares optimization using the mean centred data.
6) plot the linear model in green
7) re-plot the model centred on the original data (in yellow)

Your final plot should look something like this:



Tip: the matlab function "\" could be useful for this exercise…

EXERCISE 5 (Value: 10 marks)

The data for this exercise is contained in file “exercise5.mat”. This file contains two variables, C0 and C1, corresponding to points in classes C_0 and C_1 respectively.

In this exercise you will implement the EM algorithm for mixture of Gaussians and you will apply it to the problem of classifying a 2D point into classes C_0 and C_1. You will write one script, and possibly some function, as you will see necessary.

Having learned a model for each of the classes your code will ask the user for the coordinates of a new point and it will then provide the posterior probability for that point to belong to each of the 2 classes.

Tip: when implementing the EM algorithm, don’t forget to initialize the covariances to the covariance of a subset of your data (rather than to random values).

Marking Criteria

In order to obtain full marks for each question, you must answer it correctly and completely.
Marks will be given for writing compact, vectorised code and avoiding the use of “loops” (*for* or *while* loops) for carrying out operation on matrix and vector elements.