FGV EMAp
ESCOLA DE MATEMÁTICA APLICADA

Machine Learning 1

LAB EXERCISE 4

This lab exercise must be submitted by February 9nd, 2024 at 10:00 am.
Late submissions will not be accepted and will be marked as zero.

This lab exercise is mandatory and could be assessed.
It is worth 1% of your total final grade for this course.

Submission Instructions

Submit your files through EClass. The files you submit cannot be read by any other students. You can replace your submission as many times as you like by resubmitting it, although only the last version sent is kept.

If you have last-minute problems with your EClass submission, email your assignment as an attachment to alberto.paccanaro@fgv.br with the subject "URGENT – LAB 4 SUBMISSION". In the body of the message, explain the reason for not submitting it through EClass.

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

In this lab, you will implement a Naïve Bayes classifier and Linear Regression. For linear regression regression you will implement both the closed form solution as well as the stochastic gradient descent optimization strategy.

DATASET DESCRIPTION

You will be using 2 different datasets, that you will find on the Eclass page for this lab:

1) Car.csv: this dataset is constituted by *1728* points in 6 dimensions. Each point refers to a car, which is described by 6 attributes, all of them (first 6 columns). Each car belongs to one of possible 4 classes (last column).

   The description of the attributes is as follows

   buying: buying price
   maint : price of the maintenance
   doors : number of doors
   persons: capacity in terms of persons to carry
   lug_boot: the size of luggage boot
   safety : estimated safety of the car

   The values of the attributes are:

   Buying: v-high, high, med, low
   Maint: v-high, high, med, low
   Doors: 2, 3, 4, 5-more
   Persons: 2, 4, more
   lug_boot: small, med, big
   safety: low, med, high

   The classes and their proportions are:

CLASS	N	%
unacc	1210	70.02
acc	384	22.22
good	69	4.00
v-good	65	3.76

2) Housing.txt: this dataset is constituted by *506* points in 14 dimensions. Each point represents a house in the Boston area, and the 14 attributes that you find orderly in each column are the following:

CRIM - per capita crime rate by town
ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS - proportion of non-retail business acres per town.
CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
NOX - nitric oxides concentration (parts per 10 million)
RM - average number of rooms per dwelling
AGE - proportion of owner-occupied units built prior to 1940
DIS - weighted distances to five Boston employment centres
RAD - index of accessibility to radial highways
TAX - full-value property-tax rate per $10,000
PTRATIO - pupil-teacher ratio by town
B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
LSTAT - % lower status of the population
MEDV - Median value of owner-occupied homes in $1000's

This dataset is normally associated with 2 regression tasks: predicting NOX (in which the nitrous oxide level is to be predicted); and predicting price MED (in which the median value of a home is to be predicted).

YOUR TASKS

A) build a Naïve Bayes classifier to predict the class of the car from its attributes. Once the model will have been trained, your system will be able to take in input a new point (that was not used for training) and predict the class to which it belongs. Think about these questions:

i. what does learning mean here? There is no iterative procedure. And you don't seem to be minimizing any function. So which parameters are you really training? Can this procedure overfit?
ii. Is there an effective strategy that can be employed to avoid a NB system to predict always only one class?

B) Build a Linear Regression model. You will need to implement it twice:

i. using the Normal Equations seen in class.
ii. using stochastic gradient descent.

TIP 1: when you implement the stochastic gradient version it will be crucial to plot the Sum of Squares error as a function of the iteration.

TIP 2: you will need to initialize the weights. Start by choosing them randomly, gaussian distributed, with a very small variance…

Run some experiments with your stochastic gradient descent version, and think about these questions: what happens with different values of the step parameter? Does the system always converge to the same solution? Are you getting the same solution that you get with the closed form implementation? What happens if you initialize your weights to values that are too big?

Today you should aim at finishing a few scripts that implement the above descriptions.

During next week you should try to:

- divide your dataset into training and testing;
- measure the error on the training and testing dataset (which error measure should you use?);

- compare the change in the loss function for different values of the step parameter;
- generate descriptive plots to show your results (with title, variable names on the axes, etc).

Have fun ! ☺