Practical work 04 – 10th of October 2017
Classification Systems - Bayes

Summary for the organisation :

— Submit the solutions of the practical work before Monday 12h00 next week in Moodle.
— Preferred modality : iPython notebook.
— Alternative modality : pdf report.
— The file name must contain the number of the practical work, followed by the names of the team members by alphabetical order, for example `02_dupont_muller_smith.pdf`.
— Put also the name of the team members in the body of the notebook (or report).
— Only one submission per team.

Exercice 1 Classification system

The objective of this exercise is to build classification systems to predict whether a student gets admitted into a university or not based on their results on two exams [1]. You have historical data from previous applicants that you can use as a training set. For each training example n, you have the applicant's scores on two exams $(x_{n,1}, x_{n,2})$ and the admissions decision y_n. Your task is to build a classification model that estimates an applicant's probability of admission based on the scores from those two exams.

a. Getting started

a) Read the training data from file `ex1-data-train.csv`. The first two columns are x_1 and x_2. The last column holds the class label y.
b) Plot the training data using a scatter plot. You should get something similar to the Figure 1
c) Build a *dummy* recognition system that takes decisions randomly.
d) Compute the performance $N_{correct}/N$ of this system on the test set `ex1-data-train.csv`, with N the number of test samples and $N_{correct}$ the number of correct decision in comparison to the ground truth.

1. Data source : Andrew Ng - Machine Learning class Stanford

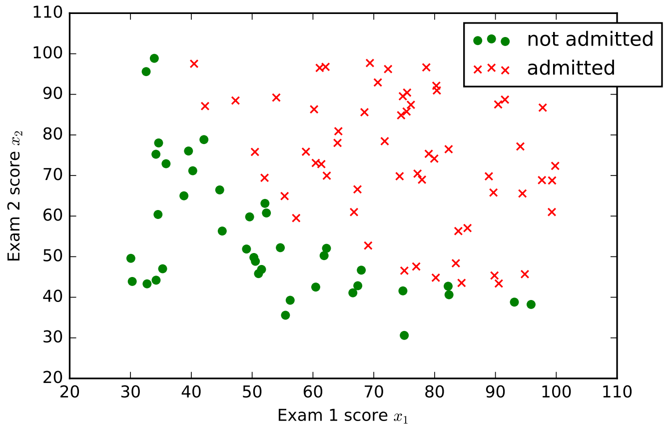

FIGURE 1 – Training data

b. K-nn classifier

Build a k-nn classifier on the data using an Euclidian distance computation and a simple majority voting criterion, i.e. decide C_0 when there is a majority of points in class 0 in the k nearest neighbours. Compute the performance of the system as a function of $k = 1 \ldots 7$. What value of k gives you the best performances ? Comment your result.

Remark : How is your system taking decisions when you have an equal number of votes for both classes with values of $k = 2, 4, 6$?

c. Bayes - Histogram

Implement a classifier based on a Bayesian approach using histograms to estimate the likelihoods.

a) Compute the priors of both classes $P(C_0)$ and $P(C_1)$.

b) Compute histograms of x_1 and x_2 for each class (total of 4 histograms). Plot these histograms. Advice : use the numpy `histogram(a,bins='auto')` function.

c) Use the histograms to compute the likelihoods $p(x_1|C_0)$, $p(x_1|C_1)$, $p(x_2|C_0)$ and $p(x_2|C_1)$. For this define a function `likelihoodHist(x,histValues,edgeValues)` that returns the likelihood of x for a given histogram (defined by its values and bin edges as returned by the numpy `histogram()` function).

d) Implement the classification decision according to Bayes rule and compute the performance of the system on the training set :
   - using only feature x_1
   - using only feature x_2
   - using x_1 and x_2 making the naive Bayes hypothesis of feature independence, i.e. $p(X|C_k) = p(x_1|C_k) \cdot p(x_2|C_k)$

Which system is the best ?

c. Bayes - Univariate Gaussian distribution

Do the same as in c. but this time using univariate Gaussian distribution to model the likelihoods $p(x_1|C_0)$, $p(x_1|C_1)$, $p(x_2|C_0)$ and $p(x_2|C_1)$. You may use the numpy functions `mean()` and `var()` to compute the mean μ and variance σ^2 of the distribution. To model the likelihood of both features, you may also do the naive Bayes hypothesis of feature independence, i.e. $p(X|C_k) = p(x_1|C_k) \cdot p(x_2|C_k)$.