NAIVE BAYES CLASSIFIER
Assignment 1 of Machine Learning I, A.Y. 2021/2022

Marco Macchia
DIBRIS - Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi
Università Degli Studi di Genova

This report describes the first assignment of Machine Learning I, what were the requirements and what are the obtained results.

I. INTRODUCTION

THE classification problem is one of the first task that a machine learning student deals with. Given a certain input, a *classifier* should associate that to a given output, i.e. a class. To do that, first the classifier needs to *study* a set of inputs including their classes. Then it take a new input, and based on the knowledge acquired during the *training session*, he categorized it.

There are many types of classifiers, one of the simplest is the **Naive Bayes Classifier**, that combined with the Laplace Smoothing gives very good results.

II. THEORY OF NBC

A. Mathematical theory

Let's call the input x (*observation*). A Classification is a rule $y()$ that, given an input, produces an output called t (*class*), so that

$$t = y(x) \tag{1}$$

In order to otuput the most appropriate class, the classifier choose the classification that has the minimum error probability. In particular, the NBC is based on a *naive* assumption, which basically isn't correct but works just the same:

$$Pr(x_1, ..., x_d|t_i) = Pr(x_1|t_i)Pr(x_1|t_i)...Pr(x_d|t_i) \tag{2}$$

The NBC classifier therefore pretends that input variables are all independent of each other.

To select the *best* class we use a *discriminant function* $g_i(x)$. This function can be compute as follows:

$$g_i(x) = Pr(t_i)[Pr(x_1|t_i)Pr(x_2|t_i)...Pr(x_d|t_i)] = \\ = Pr(t_i)\prod_{j=1}^{d} Pr(x_j|t_d) \tag{3}$$

After computing every $g_i(x)$, the classifier takes the one that has the highest value, and classifies x with t.

B. Training session

During the training session the classifier acquires data from the training set. In particular the NBC computes every conditional probability, using this formula:

$$Pr(x_j = v_k|t_i) = \frac{number\ of\ x_j = v_j\ in\ class\ t_i}{number\ of\ instance\ of\ t_i} \tag{4}$$

where v_j is the possible value for the attribute j.

C. Improvements with Laplace Smoothing

Using a small data set, some combinations that appear in the test set were not encountered in the training set, so their probability is assumed to be zero. When you multiply many terms, just a zero sets the overall results to zero.

Generally speaking, some values of some attributes could not appear in the training set, but are actually inside the entire set. The classifier should know this information, and the **Laplace Smoothing** comes in hand.

With the Laplace smoothing, a *trust factor* a is implemented in the conditioned probability computation as follows:

$$Pr(x_j = v_k|t_i) = \frac{(number\ of\ x_j = v_j\ in\ class\ t_i) + a}{(number\ of\ instance\ of\ t_i) + av} \tag{5}$$

where v is the number of possible values of the variable x.

When $a < 1$ the classifier trusts his prior belief less than the data, while with $a > 1$ the classifier trusts his prior belief more than the data. Adding a results in a probability that is never equal to zero, even if the value does not appear in the training set.

III. THE ASSIGNMENT

The given assignment consist of three tasks:
- **Task 1:** Data preprocessing
- **Task 2:** Build a naive Bayes classifier
- **Task 3:** Improve the classifier with Laplace smoothing

A. Task 1 : Data preprocessing

Usually in Matlab attributes value should be converted in positive integer values, in order to make them easier to manipulate.

The used set is a Weather data set composed by:
- 14 observations
- 4 attributes (outlook, temperature, humidity, windy)
- 2 classes (play YES, play NO)

The set is built in such a way that the first four columns are the attributes, while the last column is the target of the observation. In particular, outlook can assume three different values (*overcast, rainy or sunny*), temperature can also assume three different values (*hot, mild or cool*), humidity can assume two values (*high or normal*) and windy can assume two values (*true or false*).

B. Task 2: Build a naive Bayes classifier

The naive Bayes classifier is divided in two sections: the training section and the classification section.

Inside the training section the classifiers acquires knowledge computing every conditioned probability using the equation 4.Data are stored in a *cell matrix* in order to avoid wasting space. The cell matrix is defined and instantiated at every execution of the classifier.

After the training section the NBC classifies each observation that is inside the test set, according to the inferred rule of maximizing the discriminant function $g_i(x)$. Also if the number of columns of the test set is equal to the number of columns of the training set (i.e. ground truth are given), the error rate is computed and returned to the main function, as well as all the classifications.

The main function splits the data set in two subset, the *training subset* (70% of the entire set) and the *test subset* (the remaining 30%).The two subsets are then given as inputs to the NBC.

C. Task 3: Improve the classifier with Laplace (additive) smoothing

As seen in the equation 5, the a factor is inserted inside the equation 4. the a factor is given to the NBC as an input parameter. Apart this introduction, the classifier is just the same as before.

IV. RESULTS

In order to test and compare the behavior of the naive Bayes Classifier the classification is executed for a total of 40 iterations. In particular, 10 random subset are generated, and for each of this subset the classification is executed with different values of the factor a:

- $a = 0$ (means no Laplace Smoothing),
- $a = 1$,
- $a = 0.5$,
- $a = 2$

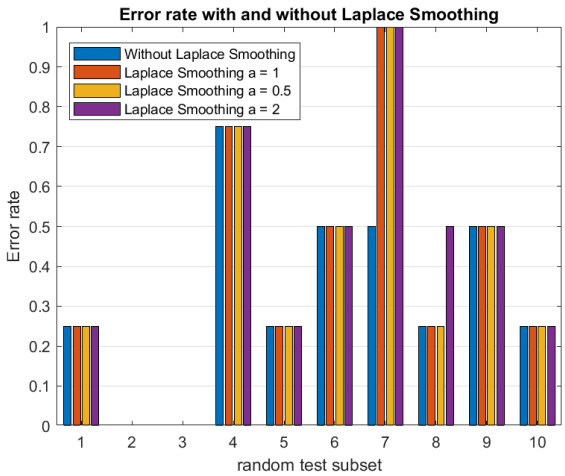

Fig. 1: Percentage of correct classifications over the test subset

For every subset and value of factor a the NBC returns to the main function the error rate of the classification. The value is computed as the number of correct classification divided by the number of observation. Notice that in this test, with only 4 observation inside the test set, the error rate can assume only five fixed values:

- $e = 0$ in case every classification is correct,
- $e = 0.25$ if there is only one wrong classification,
- $e = 0.5$ if there are two wrong classifications,
- $e = 0.75$ if there are three wrong classifications,
- $e = 1$ in case every classification is incorrect.

As shown in the image 1, the test results are quite different, and they depends on the specific train set: with 10 random subset there are some cases when the error is null (like subset 2 and 3), while there are cases where the error rate equals 1 (as with subset 7), meaning that all classifications are wrong.

Also the bar graph in figure 1 shows that the Laplace Smoothing doesn't change the result of the classification: only with subset 7 the graph shows some differences.

All the classifications are executed with a training set made of 10 observation and a test set made of 4 observation. With two larger sets the Laplace could be more incisive in the classification process.