**Università degli Studi di Genova**

**Robotics Engineering**

**Sara Romano**

**Report of Machine Learning course no.1**

# Navie Bayes Classifier

**October 2020**

# Contents

# Chapter 1

# Introduction

Naive Bayes Classifier is used for classification problems (assigning class labels to problem instances, represented as vectors of predictor values, where the class labels are defined from some finite set) and it is a supervised machine learning algorithm. The important principle of this algorithm assumes that all values of all possible predictor vectors are independent to each other.

## 1.1 Bayes Theorem

Bayes theorem provides a way of calculating the posterior probability $P(c|x)$, starting from $P(c)$, $P(x)$ and $P(x|c)$

where:

$P(c)$ = priori probability of class levels;

$P(x)$ = priori probability of each feature (or predictor);

$P(x|c)$ = likelihood which is the probability of predictor given class.

Then the Bayes Theorem is the follow:

$$P(c|x) = \frac{P(x|c) * P(c)}{P(x)}$$

Taking into account the class conditional independence:

$$P(c|X) = P(x_1|c) * P(x_2|c) * ... * P(x_n|c) * P(c)$$

Considering n = number of predictors.

Rovetta A.y. 2020/2021 *Naive Bayes classifier* 2020

# Chapter 2

# Description

It is required to compute a program capable of implementing the Naive Bayes Classifier which will explore a data set with a different number of features and classes.

The work was done on MATLAB environment. 2020

The folder ”matlabscript” contains all file.m:

1) ”data.txt”;

2) ”NavieBayesClassifier.m”;

3) ”createset.m”;

4) ”probability.m”;

5) ”classifyTestSet.m”;

6) ”probabilityLaplace.m”.

## 2.1 ”data.txt”

It is a txt file that contain the data set. It is the input of the main code program. It is a table in which the number of rows is the number of instances/observations, the number of columns is the number of predictors plus the target. The data is composed by number from 1 to 3: these indicate the number of possible levels of each feature and target. (EX: the first column contains 3 possible values which are the three levels of the first feature).

## 2.2 ”NavieBayesClassifier.m”

It it the main code.

### 2.2.1 Task 1: Data Preprocessing

The main code loads the data set and, calling a function ”createset.m”, the training set and test set are defined choosing some observations randomly (10 for training and 4 for testing).

### 2.2.2 Task 2: Build a naive Bayes classifier

The program checks that all values on the data set is lower than 1 and that the number of columns of the test set is at least the number of columns of the training set - 1.

Considering the training set, it uses the function ”probability.m” to compute the prior probability of each class and the Likelihoods of each feature respect to each class.

Once we have obtained all matrices we need, the second function ”classifytestset.m” is called for classifying the test set according to the inferred rule, and return the classification.

The comparison between the real target and the predicted target of all observations of the test set and the error rate is computed as well.

### 2.2.3 Task 3: Improve the classifier with Laplace (additive) smoothing

The training set is improved adding one row more which contains the number of levels of each feature.

The function ”probabilityLaplace.m” is called. It computes new likelihoods values considering the Laplace smoothing.

Suddenly all steps for classifying the test set are computed as well as in the task 2.

## 2.3 ”Createset.m”

This function is able to divide the data set into 2 data set: training set and test set, choosing randomly 10 observations of the first one set and the rest for the test set.

### 2.3.1 input

1) dataset.

### 2.3.2 output

1) indexes (a structure which contains the number of rows and columns of the data set, and the row indexes chosen randomly for creating the training set);

2) set (a structure which contains the two set).

## 2.4 ”probability.m”

This function computes the prior probability of each class: it counts the num of instances for each level of the class and compute $P(c)$ :

$$P(c) = \frac{Num.of.appearance.of.each.level.class}{Num.of.total.observation}$$

Then it counts how many class levels are associated with a level of a predictor and calculate the likelihoods as the ratio between the number of appearances of a feature for each class over the total number of examples of that class.

$$P.likelihoods(j, s, i) = \frac{B(j.s.i)}{num.of.appearance.of.each.class.level}$$

### 2.4.1 input

1) Training set.

### 2.4.2 output

1) num (structure which contains the num of rows and columns of the training set matrix, num of class level, the number of the appearances of each class level among the observations, num of attributes and the numb of levels for each feature);

2) Target;

3) B (matrix in 3D with the numbers of appearances of a feature for each class);

4) P (structure which contains the prior probability of each class and the likelihoods).

## 2.5 ”ClassifyTestSet.m”

This function implements the Naive Bayes Classifier: it computes the Posterior Probability of all instances of the test set, according to the Bayes Theorem.

### 2.5.1 input

1) Test set;

2) Likelihood Matrix(j,s,i);

3) Num levels class;

4) Prior Probability Class.

### 2.5.2 output

1)Num.t (a structure with the num of rows and columns of the test set);

2) P.t (a structure with the predicted Probability of each feature level based on each class level, and the Posterior Probability).

## 2.6 ”ProbabilityLaplace.m”

This function is an improvement of the ”probability.m” function, therefore it has the same structure. The fundamental variation is based on the computation of the likelihoods of the features. In formula I replaced:

$$P.likelihoods(j,s,i) = \frac{B(j.s.i)}{num.of.appearance.of.each.class.level}$$

with:

$$P.likelihoods(j,s,i) = \frac{B(j.s.i) + a}{(num.of.appearance.of.each.class.level) + av}$$

where v is the possible levels of each feature.

Therefore the input and output are the same of the ”probability.m” function.

# Chapter 3

# Results

The evaluation of the given set of data is carried out running 1000 times the program in order to get 1000 different results from different training set and test set (computed randomly).

The resultant error rates averages are:

1) 0.35 without smoothing;

2) 0.40 with smoothing.

As we can note, the error averages are quite high: it is a consequence of the number of the low observations.

We must point out another important aspect related to the classification improvement by Laplace smoothing: in this case it raises the error rate average. One reason why this occurs is due to the amount of data for the Likelihoods computation: if the input training set has all the possible values of each feature, adding a Laplace smoothing means crushing the real probabilities. Indeed the improvement is required when the input set misses some values, which appear in the test set. In this last case, the Laplace smoothing will improve the results.

# References

(2020). MathWorks. URL: `https://it.mathworks.com/support/learn-with-matlab-tutorials.html` (visited on 10/20/2019).

*Naive Bayes classifier* (2020). Wikipedia. URL: `https://en.wikipedia.org/wiki/Naive_Bayes_classifier` (visited on 10/20/2020).

Rovetta, Stefano (A.y. 2020/2021). *ml-2020-21–02*.