Applied Machine Learning in Data Science (CMPSC 445 -001)

Spring 2021, Assignment -2 (Naive Bayes Classifier**)**

Due date: 03/10/2021 at 11:59 PM (Total points: 100)

The **objectives** of this project are:

- To implement the Naive Bayes Classifier algorithm.
- To learn how to assess the classification accuracy.

Description

In this assignment you will implement the Naïve Bayes Classifier algorithm. You will write a program, in Python that implements the naive Bayes classification algorithm. You should test your implementation on several datasets. The datasets will be uploaded along with this assignment to the canvas page of the course. Each dataset is partitioned into two datasets, namely the training data and the testing data. The last attribute in each dataset is a class label. Your goal is to learn a model using the training data and use the model to classify the testing data.

Approach

Recall that the Bayes theorem allows us to write the posterior probability in terms of the likelihood and prior probability. Please refer to the class notes for the formulas. One significant aspect of the naive Bayes approach is that it makes the "naive" assumption that attributes are all independent. This leads to a much simpler way of calculating the joint probability as a product of dimension-wise probabilities:

$$P(X|c_i) = P((x_1, x_2, \cdots, x_d)|c_i) = \prod_{j=1}^{d} P(x_j|c_i)$$

For numeric data, use the probability density function (pdf) for the normal distribution to return the likelihood:

$$P(x_j|c_i) = \frac{1}{\sqrt{2\pi{\sigma_j^i}^2}} \exp^{-(x_j-\mu_j^i)^2/2{\sigma_j^i}^2}$$

Where μ_j^i and σ_j^i are the mean and standard deviation for the j attribute for the data points with class label **c_i** . To get **$P(x_j \,|c_i)$,** you can use the "norm.pdf(**x_i**,m,s)" function in python, where m is the mean, and s is the standard deviation.

For categorical data, the independence assumption leads to the following direct estimate of the probability per dimension:

$$P(x_j|c_i) = \frac{\#\ of\ times\ value\ x_j\ occurs\ in\ D_i}{|D_i|}$$

Moreover, to obtain non-zero probabilities you should employ the Laplace correction, where you add a count of one to the observed counts of each value for each class.

DataSets:

Iris Dataset

The iris dataset contains 150 instances where each instance has 4 attributes (measurements) and a class label indicating the type of iris plant. The attribute information are as follows: 1.) sepal length in cm, 2.) sepal width in cm, 3.) petal length in cm, 4.) petal width in cm The original dataset has 3 classes: Iris Setosa, Iris Versicolour, Iris Virginica. For more datasets, follow this link:

http://archive.ics.uci.edu/ml/datasets.html

In this assignment, we combined 2 classes into one class, so we have a total of 2 classes only, (-1 and 1).

Your Program:

What input parameters your program should take:

Your program should accept the file names of the training data and the testing data.

Example: (**For continuous attributes**)

trainingFile = “irisTraining.txt”

testingFile = “irisTesting.txt”

(**For categorical attributes**)

trainingFile = “buyTraining.txt”

testingFile = “buyTesting.txt”

(If you implement the program to handle either dataset, you will get 80%. To achieve 100%, you need to implement the program that can handle both datasets).

Programming Language: You can implement the algorithm in any programming language you want. However, I highly recommend **Python**.

What your program should report:

Your program output should consist of the following information:

1. The **classification accuracy** on each testing dataset.
2. The number of true positives (TPs), false positives (FPs), true negatives (TNs), and false negatives (FNs).
3. The classification Precision and Recall. (Optional)

What to submit:

❖ Electronic submission (Due: 11:59:59 PM, March 10, 2021)

- Submit your homework to **Canvas**
- Your personal information at the top (i.e., name and id)
- You should submit the results on each of the testing datasets in a **pdf file.**
- Also, submit your python codes (separate file .py)