Introduction to Machine Learning: Assignment #1

Submission date: 01\05\2023, 23:55.
Please submit the assignment via Moodle.
For questions regarding the assignment please contact:
Sharon Rotgaizer (srotgaizer@gmail.com) or
Gili Goldin (gili.sommer@gmail.com).

Assignment Instructions:

- Submissions in ***pairs***.
- The code must be written in Python 3.6 or higher.
- The code must be reasonably ***documented***.
- Please hand in one single ***.zip*** file containing ***only*** the python and pdf file with the ***exact*** name id1_id2.zip

Assignment Topics:

- Risk minimization
- Bayes rule
- MLE
- Naïve Bayes

Good Luck.

Problem 1 (20 points):

You need to arrive at a decision on which TV set to choose, based on the quality of the picture it shows.
All TVs in the market are classified as follows: bad=0.2, fair=0.5, and good 0.3. The pictures in different TV sets are distributed as follows:

	Good	Fair	Bad
sharp picture	0.9	0.5	0.2
diminished picture	0.1	0.5	0.8

The loss function is defined as follows:

$\lambda(\alpha_i \mid w_j)$	Good	fair	Bad
Buy a TV	0	5	20
Don't buy a TV	10	5	0

Make the optimal decision and provide the conditional risk associated with your decision.

Moreover, compute the total risk.

Problem 2 (20 points):

You are to be tested for a disease that has a prevalence in the population of 1 in 1000. The lab test used is not always perfect: It has a false-positive rate of 1%. [A false-positive result is when the test is positive, although the disease is not present.]. The false-negative rate of the test is zero. [A false negative is when the test result is negative while in fact, the disease is present.]

a) If you are tested and you get a positive result, what is the probability that you actually have the disease?
b) If you decide to get tested, is it more probable that you have the disease or that you don't?
c) Would the answers to (a) and/or (b) differ if you use a maximum likelihood versus a maximum a posteriori hypothesis estimation method? Comment on your answer.

Problem 3 (20 points):

In a casino, two differently loaded but identically looking dice are thrown in repeated runs.
The frequencies of numbers (number of times each number has been observed) observed in 40 rounds of play are as follows:

Dice 1, [Nr, Frequency]: [1,5], [2,3], [3,10], [4,1], [5,10], [6,11]
Dice 2, [Nr, Frequency]: [1,10], [2,11], [3,4], [4,10], [5,3], [6,2]

a) Denote by $P_i(j)$ the probability of getting the number j (for every $j \in \{1, \cdots, 6\}$) using the dice i (for every i$\in \{1,2\}$)).

   Use MLE to estimate the priors $P_i(j)$ for every pair of i, j.

b) Sometime later, one of the dice disappeared. You (as the casino owner) need to find out which one. The remaining one is now thrown 40 times and here are the observed counts: [1,8], [2,12], [3,6], [4,9], [5,4], [6,1].

   Use a Bayes' rule to decide the identity of the remaining dice.

Problem 4 (40 points):

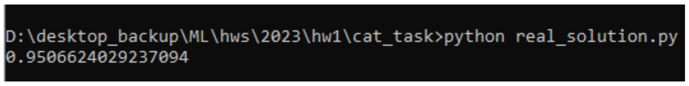
In this problem, you'll implement a basic Naïve Bayes classifier, and use it to classify sentences into 8 categories (but could be any number).

Instructions:

a. Write a function **`Pw, P = learn_NB_text()`** that computes and returns the probabilities:
   Pw - a matrix of class-conditional probabilities, $p(x|w_i)$
   P - a vector of class priors, $p(w_i)$

b. Write a function **`suc = ClassifyNB_text(Pw, P)`** that classifies all documents from the test set and computes the success rate (**`suc`**) as a number of correctly classified documents divided by the number of all documents in the test set.

c. Report the classification rate in your pdf ***<u>and</u>*** in your python script, like here (this of course, just an example, not the real expected accuracy).



```
D:\desktop_backup\ML\hws\2023\hw1\cat_task>python real_solution.py
0.9506624029237094
```

Note: Multiplying lots of probabilities, which are between 0 and 1, can result in floating-point underflow. Since $\log(xy) = \log(x) + \log(y)$, it is better to perform all computations by summing logs of probabilities rather than multiplying probabilities. Class with highest final un-normalized log probability score is <u>still</u> the most probable.

Data and Supplied Code: `textClassif.zip` contains the following files:

r8-test-stemmed.txt --test set for 8 categories.
r8-train-stemmed.txt -- train set for 8 categories.

readTrainData.py -- a python function that reads all documents from the train set and returns the following data structures:

- texAll - list of documents; each entry corresponds to a document which is list of words.
- lbAll list of documents' labels.
- voc - set of all distinct words in the train set.
- cat - set of document categories.

Instructions for using the code:

1. Unzip the data and the code in the same directory.

2. Run: `texAll, lbAll, voc, cat = readTrainData('r8-train-stemmed.txt')`

3. Write the required functions and run them using these data structures. You can use the code in readTrainData.py as an example for reading the test data.

4. Report the success rate on the test set.

- The solution should be formatted as a report and running code should be included in a digital form.

- The code must be reasonably documented.

Tips:

1. If you decide to keep the CSV files in a separate folder, make sure you use os.path.join in order to get a platform-specific path to them. **Using slashes (/ or \) directly might lead to problems running the code on different operating systems.**

2. Try to keep the code as clean, concise, and short as possible. You shouldn't need more than 100-120 lines of code.

3. Do NOT use any out-of-the-box implementation of Naïve Bayes (scikit-learn etc.). Reuse is usually encouraged, but in this case, you'll need to implement it yourself. **Any submission using a premade implementation of the classifier will not get any points!**

4. Submit your Python code as a .py file, in addition to your answers for the theoretical questions.

5. Your submission must be entirely your own. **Any attempts of plagiarism will lead to disciplinary actions. Using ChatGPT is not just forbidden, but can be easily detected and counted as plagiarism.**

Problem 5 (5 points bonus):

We use Bayesian decision rule to classify samples and take a decision, such that minimizes the risk.

To minimize the risk, we use some loss function $\lambda(a_i|c_j)$ and take the decision α_i that minimizes $R(\alpha_i|x)$.

We now define the 0-1 loss function, $\lambda(a_i|c_j) = \begin{cases} 1 & i \neq j \\ 0 & i = j \end{cases}$.

Prove that under this loss function, the Bayesian decision rule is equivalent to the MAP classification.