CSCI 4360/6360 Data Science II
Department of Computer Science
University of Georgia

Assignment 1: Machine Learning Review

DUE: Tuesday, September 3 by 11:59:59pm

Out August 20, 2019

Overview

1 Submitting

All submissions will go to **AutoLab**. You can access AutoLab at https://autolab.cs.uga.edu. You will first need to create an account on AutoLab before you can be added to the `csci4360-fa19` course and submit assignments.

You can submit deliverables to the **Assignment 1** assignment that is open. When you do, you'll submit two files:

1. `assignment1.py`: the Python script that implements your algorithms, and
2. `assignment1.pdf`: the PDF write-up with any questions that were asked

DO NOT DEVIATE from these naming conventions! Doing so will result in the autograder issuing failing grades!

There's no penalty for submitting as many times as you need to, but keep in mind that swamping the server at the last minute may result in your submission being missed; AutoLab is programmed to close submissions *promptly* at 11:59pm on September 3 so give yourself plenty of time! A late submission because the server got hammered at the deadline will *not* be acceptable (there is a *small* grace period to account for unusually high load at deadline, but I strongly recommend you avoid the problem altogether and start early).

2 Reminders

- If you run into problems, ping the `#questions` room of the Slack chat. If you still run into problems, ask me. But please please please, **do NOT** ask Google to give you the code you seek! I will be on the lookout for this (and already know some of the most popular venues that might have solutions or partial solutions to the questions here).
- Prefabricated solutions (e.g. `scikit-learn`) are NOT allowed! You have to do the coding yourself!
- If you collaborate with anyone, just mention their names in a code comment at the top of your homework file.

Questions

1 Conditional Probability and the Chain Rule **[10pts]**

[5pts] Recall the definition of conditional probability:

$$P(A|B) = \frac{P(A \cap B)}{P(B)},$$

where $\cap$ means "intersection". Prove that $P(A \cap B \cap C) = P(A|B, C)P(B|C)P(C)$.

[5pts] Derive Bayes' Theorem from the law of conditional probability, and define each term in the equation with a 1-sentence description.

2 Total Probability **[10pts]**

Let's say I have two six-sided dice: one is fair, one is loaded. The loaded die has:

$$P(x) = \begin{cases} \frac{1}{2} & x = 6 \\ \frac{1}{10} & x \neq 6 \end{cases}$$

In addition to the two dice, I have a coin which I flip to determine which dice to roll. If the coin flip ends up heads I will roll the fair die, otherwise I'll roll the loaded one. The probability that the coin flip is heads is $p \in [0, 1]$.

[5pts] What is the expectation of the die roll, in terms of p?

Hint: Recall that the expected value $E[X]$ of a discrete random variable X (e.g., a coin flip) can be computed as

$$E[X] = \sum_i x_i P(X = x_i)$$

[5pts] What is the variance of the die roll, in terms of p?

Hint: Recall that the variance $\text{Var}(X)$ of a random variable X can be computed as

$$\text{Var}(X) = E[X^2] - (E[X])^2$$

3 Naive Bayes **[15pts]**

Consider the learning function $f(X) \rightarrow Y$, where class label $Y \in \{T, F\}$ and $X = \{x_1, x_2, ..., x_n\}$, where x_1 is a boolean attribute and $x_2, ..., x_n$ are continuous attributes.

[10pts] Assuming the continuous attributes are modeled as Gaussians, give and *briefly* explain the total number of parameters that you would need to estimate in order to classify a future observation using a Naive Bayes (NB) classifier.

Hint: recall that a Naive Bayes classifier requires both the conditional probabilities $P(X = x_i|Y)$ and the class prior probability $P(Y)$.

[5pts] How many more parameters would be required without the conditional independence assumption? No need for an exact number; an order of magnitude estimate will suffice.

4 Logistic Regression **[15pts]**

In Logistic Regression (LR), we assume the observations are independent of each other (not *conditionally* independent, just independent).

[10pts] Prove the decision boundary for Logistic Regression is linear. i.e., show that $P(Y|X)$ has the form:

$$w_0 + \sum_i w_i X_i,$$

where $Y \in \{0, 1\}$, and the quantity of the sum in the above equation will determine whether LR predicts 1 or 0.

Hint: Recall that

$$P(Y = 0|X) = \frac{1}{1 + \exp(w_0 + \sum_i w_i X_i)},$$

and that $P(Y = 0|X) + P(Y = 1|X) = 1$.

[5pts] *Briefly* describe one advantage and one disadvantage of LR compared to NB.

5 Coding **[50pts]**

In this problem you will implement Logistic Regression (LR) for a document classification task.

[10pts] Imagine a certain word is never observed during training, but appears in a testing set. What will happen when the NB classifier predicts the probability of the word? Explain. Will LR have the same problem? Why or why not?

[40pts] Implement LR in a script named `assignment1.py`. This script should accept three arguments, in the following order:

1. a file containing training data
2. a file containing training labels
3. a file containing testing data

Your script should be able to be invoked as follows:

```
> python assignment1.py train.data train.labels test.data
```

For training LR, we found a step size η around 0.0001 worked well.

The data files (`train.data` and `test.data`) contains three numbers on each line.

```
<document_id> <word_id> <count>
```

Each row of the data files contains the count of how often a word (identified by ID) appears in a certain document. The corresponding label file for the training data has only one number per row of the file: the label, 1 or 0, of the document in the same row of the data file.

For each line in the testing file, your code should print a predicted label (0 or 1) by itself on a single line. These output will be used to autograde your LR implementation on AutoLab. For example, if the following `test.data` file has four lines (words) in it, your program should print out four lines, each with either a 0 or a 1, e.g.

```
> python assignment1.py train.data train.labels test.data
0
1
1
0
```

Don't be alarmed if the training process of LR takes a few minutes; a good sanity check is to make sure your weights are changing on each iteration. It is **highly recommended** that you use NumPy vectorized programming to train the weights efficiently.