

Homework 1: Machine Learning Review

DUE: Tuesday, January 28 by 11:59:59pm

Out January 9, 2025

Overview (read this first!)

1 INTRODUCTION

Welcome to the first homework assignment of the semester! This is meant to introduce you to how the rest of the assignments will go, and hopefully work out any wrinkles in the process. Some key guidelines to follow:

- **Read all instructions.** I add/modify/clarify assignments every semester in response to issues the previous time I teach it, so in theory most of the major problems you'll run into should be addressed by following the instructions.
- **Ask if you run into problems.** If, after reading everything, you're still running into problems, ask me! It's what I'm here for!
- **Start early, don't panic.** At the very least, read through the whole assignment the day it comes out, so you have some idea of what work it entails. HW1 in particular usually goes over like a lead balloon, partly because it's the first, but also because it doesn't have any extra credit (unlike pretty much all the other assignments). Do your best, ask questions, ping the Discord, and I bet you'll do better than you think.

2 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 `csci-x360-sp25` course and submit assignments. If you're having trouble accessing AutoLab, try these (in order):

1. Are you trying to connect from the campus wifi or campus network? OR, are you connected to the campus VPN? If the answer to both these questions is "no", then you won't be able to access AutoLab.
2. Have you created an AutoLab account? If not, create one first.
3. Once you've created an account, **let me know** so I can activate it for the course.
4. After confirming steps 1-3 have been completed successfully, if you are still having trouble connecting to AutoLab to submit your assignment, please post in `#techprobs` ASAP.

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

- `homework1.py`: the Python script that implements your algorithms, and
- `homework1.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 January 28 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).

To create the tarball archive to submit, run the following command (on a command prompt / terminal):

```
tar cvf homework1.tar homework1.py homework1.pdf
```

This will create a new file, `homework1.tar`, which is basically a zip file containing your Python script and PDF write-up. Upload the tar archive to AutoLab.

Also, to save time while you're working on the coding portion, you are welcome to create a tarball archive of just the Python script and upload that to AutoLab. Once you get the autograder score you're looking for, you can then include the PDF in the folder, tarball everything, and upload it. AutoLab stores the entire submission history of every student on every assignment, so your autograder (code) score will be maintained and I can just use your most recent submission to get the PDF.

3 REMINDERS

- If you run into problems, ping the **#questions** room of the Discord chat.
- Please **do NOT** ask Google! I will be on the lookout for this (and already know some of the most popular venues for solutions to the questions here).
- If AutoLab hiccups (an error message, an autograder failure you're CONVINCED isn't because of your code), ping the **#techprobs** Discord channel.
- Prefabricated solutions (e.g. **scikit-learn**) are NOT allowed! You have to do the coding yourself!
- If you collaborate with anyone or anybot, please mention their names in a code comment and/or at the top of your homework writeup.

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 $X = \{x_1, x_2, \dots, x_n\}$, where x_1 is a boolean attribute and x_2, \dots, 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 (two sentences total is plenty).

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 `homework1.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 homework1.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 homework1.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 (this can be a simple `print`

statement). It is **highly recommended** that you use NumPy vectorized programming to train the weights efficiently.

Once you've tuned your script so it trains correctly and spits out a reasonable testing accuracy (should be substantially above random chance), give it a try on AutoLab! Just follow the submission instructions, and check your score on the scoreboard. Good luck!