CS 274A Homework 5

Probabilistic Learning: Theory and Algorithms, CS 274A, Winter 2024

Due: 3pm, Wednesday March 6th, submit via Gradescope

Instructions and Guidelines for Homeworks

- Please answer all of the questions and submit your solutions to Gradescope (either hand-written or typed are fine as long as the writing is legible).
- All problems are worth equal points (10 points) unless otherwise stated. All homeworks will get equal weight in computation of the final grade for the class (with lowest-scoring homework being dropped).
- The homeworks are intended to help you better understand the concepts we discuss in class. It is important that you solve the problems yourself to help you learn and reinforce the material from class. If you don't do the homeworks you will likely have difficulty in the exams later in the quarter.
- In problems that ask you to derive or prove a result you should submit a complete mathematical proof (i.e., each line must follow logically from the preceding one, without "hand-waving"). Be as clear as possible in explaining your notation and in stating your reasoning as you go from line to line.
- If you can't solve a problem, you can discuss the high-level concepts *verbally* with another student (e.g., what concepts from the lectures or notes or text are relevant to a problem). However, you should not discuss any of the details of a solution with another student. In particular, do not look at (or show to any other student) *any written material* directly related to the homeworks, including other students' solutions or drafts of solutions, solutions from previous versions of this class, etc. The work you hand in should be your own original work.
- If you need to you can look up standard results/definition/identities from textbooks, class notes, textbooks, other reference material (e.g., from the Web). If you base any part of your solution on material that we did not discuss in class, or is not in the class notes, or is not a standard known result, then you may want to rovide a reference in terms of where the result is from, e.g., "based on material in Section 2.2 in ....." or a URL (e.g., Wikipedia).

Required Reading for Homework 5:

1. NoteSet 6 on Classification (on class Website)

Problem 1: Bayes Error Rate with Uniform and Exponential Densities

The Bayes error rate is defined as the optimal (minimum achievable) error rate for a classification problem. It can be expressed as a sum of integrals, where the sum is over different decision regions, such that within each decision region one of the classes is more likely than any other. See Section 6 in NoteSet 6 for full details.

Consider the classification problem with a 1-dimensional feature x, where $p(x|y = 1)$ is a uniform density $U(a, b)$, and where $p(x|y = 0)$ is an exponential density $\lambda \exp(-\lambda x)$ with $\lambda = 1$ (for $x \geq 0$, with $p(x|y = 0) = 0$ for $x < 0$. Given this setup answer the following problems.

1. Assume $a = 0, b > 0$. Derive a general equation (or set of equations) that defines the decision boundary (or boundaries) as a function of the parameters.
2. Let $a = 2, b = 4, \lambda = 1$. Assume $p(y = 1) = 0.5$. Generate a plot for each of $p(x|y = 1)p(y = 1)$ and $p(x|y = 0)p(y = 0)$ (put them on the same plot) and show clearly the location of the decision region or regions. You can let the x-axis range from 0 to 8.
3. Compute the Bayes error rate for part 2 of this problem to an accuracy of within 3 decimal places. Show clearly how you computed the Bayes error rate.

Problem 2: Decision Boundaries and Bayes Error Rate for 1d Gaussians

Consider a classification problem with a 1-dimensional feature x, where $p(x|y = 1)$ is a Gaussian density with known mean μ_1 and variance σ_1^2, and where $p(x|y = 0)$ is a Gaussian density with known mean μ_0 and variance σ_0^2. Assume that $p(y = 1)$ is also known. Given this setup answer the following problems.

1. Derive a general equation that defines the decision boundary (or boundaries) as a function of the parameters above. Note that for a 1-dimensional problem that the decision boundary (or boundaries) will be in the form $x^* = \ldots$ where the left-hand side is a decision boundary and the right-hand side is some function of the parameters of the problem.
2. Let $\mu_1 = 0, \sigma_1^2 = 1, \mu_2 = 3, \sigma_2^2 = 3$. Assume that $p(y = 1) = 0.5$. Generate a plot for each of $p(x|y = 1)p(y = 1)$ and $p(x|y = 0)p(y = 0)$ (put them on the same plot) and show clearly the location of the decision region or regions. You can let the x-axis range from -8 to 8.
3. Compute the Bayes error rate for part 2 of this problem to an accuracy of within 3 decimal places. Show clearly how you computed the Bayes error rate.

Problem 3: Nearest-Mean Classifier

Consider a two-class classification problem with a d-dimensional feature vector $\underline{x}$ and $y \in \{0, 1\}$. Assume we know the mean vector μ_1 for the data from class 1, and also the mean vector μ_0 for the data from class 2 (e.g., we might have estimated these two means from a large set of labeled training data; in this problem you can treat the two means as known).

A very simple classifier is the nearest-mean classifier (or "template classifier") which operates as follows: compute the Euclidean distance between $\underline{x}$ and $\underline{\mu}_1$ and the Euclidean distance between $\underline{x}$ and $\underline{\mu}_0$, and then predict the class corresponding to the smaller of the two distances (if they are equal then toss a coin).

1. Prove that this nearest-mean classifier is in general equivalent to a Gaussian classifier with specific restrictions on its covariance parameters. Show clearly how you derived your results. A Gaussian classifier for this problem has Gaussian parameters $\underline{\mu}_1, \Sigma_1$ for $p(\underline{x}|y = 1)$, and $\underline{\mu}_0, \Sigma_0$ for $p(\underline{x}|y = 0)$. You can assume $\underline{\mu}_1$ is the same for both the Gaussian and the nearest-mean classifier; and that $\underline{\mu}_0$ is also the same for both. Also assume $p(y = 1) = 0.5$ in this part of the problem. (Hint: to start this problem, you can define what the general form is for a decision boundary for a two-class multivariate Gaussian classifier and work from there).

2. Now say $0.5 < p(y = 1) < 1$. Keeping the same restrictions as you derived in part 1 for the covariance parameters, show clearly how the fact that $0.5 < p(y = 1) < 1$ leads to a simple modification of the nearest-mean classifier. Define (e.g., with an equation) how you would make a prediction for an input $\underline{x}$ using this "modified nearest-mean classifier" and explain clearly how you arrived at your result.

Problem 4: Conditionally Independent Experts

Let Y be a binary class variable taking values $y \in \{0, 1\}$. Let X_i be a feature taking feature values x_i (potentially vector-valued) with $i = 1, \ldots, M$. Associated with each of the M features X_i is an "expert" that given a feature value x_i produces a prediction $P(y = 1|x_i)$. Individual experts could for example correspond to machine learning models or humans.

Now consider a decision-maker that wishes to compute $P(y = 1|x_1, \ldots, x_M)$ but that doesn't know the x_i values directly. Instead the decision-maker is only given the expert predictions $P(y = 1|x_i), i = 1, \ldots, M$. In addition the decision-maker knows the marginal probability $P(y = 1)$. This could model a situation for example where there are multiple different pieces of medical information x_i relevant to predicting disease status Y for a patient, but the information x_i can't be provided to the decision-maker for privacy reasons—however, all the individual expert predictions $P(y = 1|x_i)$ can be provided.

We will analyze the case where the decision maker assumes that the X_i variables are conditionally independent given Y. Also say that the decision maker assumes that each expert i is providing the true probability $P(y = 1|x_i)$ rather than an estimate of this probability.

1. Derive an equation that shows how the decision-maker can compute the odds, $\frac{P(y=1|x_1,\ldots,x_M)}{P(y=0|x_1,\ldots,x_M)}$ based on the information provided above.

2. Show that the log-odds in part (1) can be written as a linear function of the log-odds from the individual experts, plus an additional term that depends on the marginal probability of Y.

3. Interpret your result for the case $M = 1$ and explain in words what is qualitatively different to the case for $M > 1$.

4. There are multiple other ways the decision-maker could combine information from the M experts (such as averaging the predictions or using voting). For example, say the decision-maker were to threshold the individual probabilities of each expert, i.e., $z_i = 1$ if $P(y = 1|x_i) \geq 0.5$ and $z_i = 0$ otherwise (so the z_i in effect correspond to the votes of individual experts), and the decision maker then computes $P(y = 1|x_1, \ldots, x_M) \approx \frac{1}{M}\sum_i z_i$, i.e., takes the average of the votes. Provide an example for $M = 3$ that shows that illustrates clearly why this strategy of combining information (given the assumptions above) is suboptimal compared to the solution you derived in part 1 above.

Problem 5: Bayesian Estimation for Logistic Classifiers (with Python Notebook) (20 points)

This problem will involve the use of a Python notebook which is running in Google Colab.

Before you start working on the solution to the problems below, please make sure to do the following first:

- Read Section 10.5 (up to end of Section 10.2.5.1) in Kevin Murphy's book: Probabilistic Machine Learning: An Introduction, March 2022, available online at `https://probml.github.io/pml-book/book1.html`. This will give you the relevant math background for the problem.

- In your browser open up the notebook for this problem: `https://colab.research.google.com/drive/1sLoNn3Il6_DrYcISE1KH-8IMfcGPzIHh#scrollTo=BlpNXwgrpBEb`. I recommend you make a local copy (e.g., in the menu at the top click "File: Save to Drive"). This notebook is running in Google Colab. If you are not familiar with Google Colab you may want to first read an introductory tutorial on Jupyter notebooks and Colab, or have someone show you how it works.

- Confirm that you can run all of the cells in the notebook in order and that various graphs are generated. Don't change any of the code.

The notebook is going through the following steps:

- We generate data in a 2-dimensional space x_1, x_2 for a two-class problem, where the class label $y \in \{0, 1\}$. The data in each class is Gaussian, i.e., $p(x|y = 1)$ and $p(x|y = 0)$ are each Gaussians, with the same covariance structure and different means.

- We then investigate fitting a logistic classifier to this data of the form $p(y = 1|x) = \frac{1}{1+exp(-w_1x_1-w_2x_2)}$. Note that we don't fit an intercept term here (which means it is being set to 0 in effect): this restriction allows us visualize likelihoods, etc, in the two-dimensional (w_1, w_2) space.
- We define relatively weak Gaussian priors over (w_1, w_2), with mean zero and standard deviation equal to α in the priors. (Its defined as a joint prior, but the covariance is diagonal, so in effect its the product of two independent priors).
- The notebook then plots the surface (as contours) of the log-likelihood and the unnormalized posterior, $\log p(D|\theta)p(\theta)$,in the two-dimensional space defined by w_1, w_2.
- It then estimates (by finding the maximum of $\log p(D|\theta)p(\theta)$) the MAP values of w_1, w_2.
- Since the posterior cannot be computed in closed form for a logistic model, the notebook computes a Laplace approximation to the true posterior, in the form of a Gaussian density: see Section 4.6.8.2 in the Murphy book for details on the Laplace approximation, in particular Equation 4.214).
- Finally, we can approximate the posterior predictive for the weights, by sampling w_1, w_2 values from the approximate Laplace posterior, and then averaging over these samples to compute $p(y = 1|x)$ for any particular input x. (See Section 10.5.2 and 10.5.2.1 for details).

You should go through the notebook at this point and make sure you can see in the code where all of the steps above are happening. Note that even if you don't know Python, the syntax is fairly simple and with the comments you should be able to figure out how the steps above map to the different cells of code and the graphs generated in the notebook.

Answer the following questions and submit your answers (no need to submit any code or graphs). Note that it may be helpful to print out values of certain variables in the code, e.g., if you want to print a number w just add a line with `print(w)` in the code. Feel free to use equations as well as words in your answers.

1. The datapoints (if you use the provided random seed) are linearly separable (i.e., we can draw a linear decision boundary to perfectly separate the data from the two classes). Explain in 1 or 2 sentences what this implies for generating a maximum likelihood estimate for the logistic model. You can use the plot of the log-likelihood surface in the notebook to support your answer.
2. Why will adding a prior help with any issues that may come up with maximum likelihood in part 1 of this question?
3. What are the values of the MAP estimates for w_1, w_2?
4. Comment on the shape of the Laplace approximation relative to the shape of the unnormalized posterior (both are generated as plots by the notebook).
5. Comment on the shape of the posterior predictive for $p(y = 1|x)$ versus the shape $p(y = 1|x, w_{MAP}$ (both are shown as contour plots in the notebook)[1]. Explain as best you can (in 2 or 3 sentences or equations) what the specific differences are and why these differences exist.

[1]Technically $p(y = 1|x)$ should be written in full as $p(y = 1|x, D_x, D_y,$ prior params)

6. What is the true optimal decision boundary for these two Gaussian classes i.e., the optimal values for w_1, w_2 if we had an infinite amount of data? (this requires solving for the decision boundary as in problems 1 and 2).

7. If the amount of training data were to approach infinity, could the logistic model perfectly match the optimal decision boundary from part 6? Explain your answer in 1 or 2 sentences.