Homework Assignment # 2

Due: Tuesday, Feb. 21, 2023, 11:59 p.m. Mountain time
Total marks: 100

Question 1. [25 MARKS]

Imagine that you would like to predict if your favorite table will be free at your favorite restaurant. The only additional piece of information you can collect, however, is if it is sunny or not sunny. Therefore, you would like to predict whether the table will be free or not given the weather.

You collect paired samples from visit of the form (is sunny, is table free), where it is either sunny (1) or not sunny (0) and the table is either free (1) or not free (0). Your goal is to learn P(Table is Free|Weather Information), which then also automatically tells you P(Table is Not Free|Weather Information) by taking 1-P(Table is Free|Weather Information). Given these learned probabilities, you can make predictions about whether your table is free or not by checking if P(Table is Free|Weather Information) > 0.5. If there is a greater than 50% probability your table is free, you are happy to risk the icy streets to get to your favorite restaurant.

Hint: To help you with this question, see Section 5.4, Example 22 about books.

(a) [10 MARKS] Formulate this distribution learning problem as a maximum likelihood problem. Start by stating what the random variables are for this problem and what values they can take. Then explain what distributions on these random variables you want to learn and what parameters need to be learned. Finally, write the log likelihood explicitly for those distributions and parameters, $p(\mathcal{D}|\mathbf{w})$ where $\mathcal{D} = \{\{x_i, y_i)\}_{i=1}^n$ is a dataset of the collected paired samples and $\mathbf{w}$ are your parameters. The parameters are bold because they might consist of more than one parameter, and so would be a vector rather than a scalar. **Note:** You do not need to solve this maximum likelihood problem, just formalize it.

(b) [10 MARKS] Assume you have collected data for the last 10 days and computed the maximum likelihood solution $\mathbf{w}^*$ to the problem formulated in (a). You do not actually have to compute the solution, just assume that you did and now have estimated the parameters $\mathbf{w}^*$ for your distribution. If it is sunny today, then how would you predict if your table will be free? Be precise and explain how you would use your learned distribution and parameters to do this.

(c) [5 MARKS] Imagine you could further gather information about if it is morning, afternoon, or evening. How does this change the maximum likelihood problem? You do not need to write the log likelihood explicitly for this question, just explain if you need any new random variables, what values it can take, what the distribution is, and what parameters you will need.

Question 2. [20 MARKS]

We can combine simple distributions to produce more complex (multi-modal) distributions using *mixtures*. The below figure shows what occurs if we take a convex combination of two Gaussians. We can write the distribution for such a random variable X with density corresponding to an equal mixture of two Gaussians, with unknown means μ_1, μ_2 and unknown variances σ_1^2, σ_2^2, as follows.

$$p(x|\mathbf{w}) = 0.5\mathcal{N}(x|\mu_1, \sigma_1^2) + 0.5\mathcal{N}(x|\mu_2, \sigma_2^2) \tag{1}$$

where we write the four parameters $\mathbf{w} = (\mu_1, \mu_2, \sigma_1, \sigma_2)$ and

$$\mathcal{N}(x|\mu, \sigma^2) = (2\pi)^{-1/2}\sigma^{-1}\exp(-(x-\mu)^2/(2\sigma^2)). \tag{2}$$

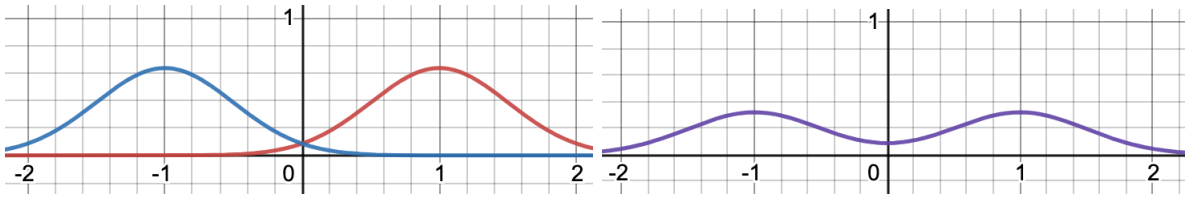

Figure 1: The blue curve is a Gaussian with $\mu_1 = -1$ and $\sigma_1 = 0.5$ and the red curve is a Gaussian with $\mu_1 = 1$ and $\sigma_1 = 0.5$. The purple curve is the mixture of the two, as in Equation (1). The purple curve allows us to model a bimodal distribution (two peaks), where now the two most likely values are -1 and 1, with density decreasing from these points. If we sample from this distribution, then we will see points centered around -1 and 1, with reasonable likelihood for point between the two (including at zero), and very low likelihood for points outside -2 and 2.

It is easy to show that $p(x|\mathbf{w})$ is a valid density, because

$$\begin{aligned}\int p(x|\mathbf{w})dx &= \int 0.5\mathcal{N}(x|\mu_1, \sigma_1^2) + 0.5\mathcal{N}(x|\mu_2, \sigma_2^2)dx \\ &= 0.5\int \mathcal{N}(x|\mu_1, \sigma_1^2)dx + 0.5\int \mathcal{N}(x|\mu_2, \sigma_2^2)dx \\ &= 0.5 + 0.5 = 1.\end{aligned}$$

You set forth to learn this distribution $p(x|\mathbf{w})$. However, now when you take the log-likelihood, you find that the log does not help as much, because the sum gets in the way of the log being applied to the exponentials.

$$\ln p(x|\mathbf{w}) = \ln\left(0.5\mathcal{N}(x|\mu_1, \sigma_1^2) + 0.5\mathcal{N}(x|\mu_2, \sigma_2^2)\right)$$

The log still helps convert the product over samples into a sum, for a given dataset of n iid samples from this distribution $\mathcal{D} = \{x_i\}_{i=1}^n$,

$$\ln p(\mathcal{D}|\mathbf{w}) = \ln \prod_{i=1}^n p(x_i|\mathbf{w}) = \sum_{i=1}^n \ln p(x_i|\mathbf{w})$$

Despite this difficulty, you are determined to learn this distribution, because you are confident it will do a better job of modeling your data. Your goal in this question is to obtain a procedure to estimate $\mathbf{w} = (\mu_1, \mu_2, \sigma_1, \sigma_2)$.

(a) [15 MARKS] Compute the gradient (partial derivatives) of your negative log likelihood objective $c(\mathbf{w}) \doteq -\ln p(\mathcal{D}|\mathbf{w})$. Start by computing the gradient of $\ln p(x_i|\mathbf{w})$. To simplify notation, consider defining $g_1(\mu_1, \sigma_1) = \sigma_1^{-1}\exp(-(x_i - \mu_1)^2/(2\sigma_1^2))$, $g_2(\mu_2, \sigma_2) = \sigma_2^{-1}\exp(-(x_i - \mu_2)^2/(2\sigma_2^2))$ and $g(\mu_1, \sigma_1, \mu_2, \sigma_2) = g_1(\mu_1, \sigma_1) + g_2(\mu_2, \sigma_2)$.

(b) [5 MARKS] Write the (first-order) gradient descent update rule for your parameters, using the gradient you compute, assuming you start from current point $\mathbf{w}_t$ and have stepsize η_t.

Question 3. [55 MARKS]

In this question, you will implement an algorithm to estimate $p(y|x)$, for a batch of data of pairs of (x, y): $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ where both $x_i, y_i \in \mathbb{R}$. We have provided you with a `julia` codebase as a `pluto notebook`, to load data and run an algorithm to estimate the parameters for $p(y|x)$. Please see this document linked here with instructions on how to get started with Julia.

It uses a simplified Kaggle data-set[1], where we use only the weight (x) and height (y) of 10,000 people. The provided data-loader function splits the data-set into a training set and testing set, after centering the variables so that the mean values for random variables x, and y will be zero. The notebook `A2.jl` has three baseline algorithms, for comparison: a random predictor, mean predictor and range predictor. These baselines are sanity checks: your algorithm should be able to outperform random predictions, and the mean value of the target in the training set. Because of randomization in some of the learning approaches, we run each of the algorithms 10 times for different random training/test splits, and report the average error and standard error, to see how well each algorithm averaged across splits.

(a) [10 MARKS] In this assignment we assume we are learning $p(y|x) = \mathcal{N}(\mu = wx, \sigma^2 = 1.0)$ for an unknown weight $w \in \mathbb{R}$. For fun, you can derive the stochastic gradient descent update for the negative log-likelihood for this problem. But, we also provide it for you here. For one randomly sampled (x_i, y_i), the stochastic gradient descent update to w is:

$$w_{t+1} = w_t - \eta \, (x_i w_t - y_i) x_i \tag{3}$$

You will run this iterative update by looping over the entire dataset multiple times. Each pass over the entire dataset is called an *epoch*. At the beginning of each epoch, you should randomize the order of the samples. Then you iterate over the entire dataset, updating with Equation (3). Implement this algorithm to estimate b, by completing the code in cell `SimpleStochasticRegressor`.

(b) [10 MARKS] Test the algorithm implemented in (a) by moving to the experiment section and running `SimpleStochasticRegressor`, along with algorithms `RandomRegressor` and `MeanRegressor`. We fixed the stepsize for `SimpleStochasticRegressor` to 0.01 and the epochs to 30. Report the average performance and standard deviation of the test error (squared errors) for each of these three approaches.

(c) [15 MARKS] Next implement a mini-batch approach to estimate w, in `MiniBatchRegressor`. The idea is similar to stochastic gradient descent, but now you use blocks (or mini-batches) of N_{batch} samples to estimate the gradient for each update. For each epoch, you iterate over the dataset in order. For the first mini-batch update, the update equation is

$$w_{t+1} = w_t - \eta g_t \qquad \text{where } w_t = \frac{1}{N_{\text{batch}}} \sum_{i=1}^{N_{\text{batch}}} (x_i w_t - y_i) x_i \tag{4}$$

Then the next update uses the next mini-batch of points, $(x_{N_{\text{batch}}+1}, y_{N_{\text{batch}}+1}), \ldots, (x_{2N_{\text{batch}}}, y_{2N_{\text{batch}}})$. In total, for one epoch, you will complete $\lceil n/N_{\text{batch}} \rceil$ updates. Make sure your code is robust to the fact that the number of samples n might not be divisible by the batch size. The very last batch could be shorter, and you should normalize that last mini-batch by the number of samples in the mini-batch rather than by N_{batch}.

(d) [10 MARKS] Next you will implement a more sensible strategy to pick stepsizes. You will

[1] `https://www.kaggle.com/mustafaali96/weight-height`

implement the heuristic for adaptive stepsizes given in the notes

$$\eta_t = (1 + |g_t|)^{-1} \tag{5}$$

where g_t is the gradient in the update $w_{t+1} = w_t - \eta_t g_t$. Implement this heuristic for SGD in `MiniBatchHeuristicRegressor`. We provide the implementation of `BatchHeuristicRegressor` that uses this same stepsize heuristic for the `BatchRegressor`. Note that both can use this heuristic, it is simply the case that the g_t is definde differently for the two functions. The g_t for `MiniBatchHeuristicRegressor` uses a minibatch and `BatchHeuristicRegressor` uses the whole dataset to compute g_t.

(e) [10 MARKS] Run the mini-batch approach and the batch approach, both with the adaptive stepsizes given by Equation (5). Again, leave the number of epochs at 30 in the provided code, and leave the stepsizes as Report and compare the test error of them before and after using adaptive stepsize selection strategies, along with the already given baselines, based on the average test error after 30 epochs. (No need to comment on the learning curve).

Homework policies:

Your assignment should be submitted as two pdf documents and a .jl notebook, on eClass. **Do not** submit a zip file with all three. One pdf is for the written work, the other pdf is generated from .jl notebook. The first pdf containing your answers of the write-up questions must be written legibly and scanned or must be typed (e.g., Latex). This .pdf should be named Firstname_LastName_Sol.pdf, For your code, we want you to submit it both as .pdf and .jl. To generate the .pdf format of a Pluto notebook, you can easily click on the circle-triangle icon on the right top corner of the screen, called Export, and then generate the .pdf file of your notebook. The .pdf of your Pluto notebook as Firstname_LastName_Code.pdf while the .jl of your Pluto notebook as Firstname_LastName.jl. All code should be turned in when you submit your assignment.

Because assignments are more for learning, and less for evaluation, grading will be based on coarse bins. **The grading is atypical**. For grades between (1) 80-100, we round-up to 100; (2) 60-80, we round-up to 80; (3) 40-60, we round-up to 60; and (4) **0-40, we round down to 0**. The last bin is to discourage quickly throwing together some answers to get some marks. The goal for the assignments is to help you learn the material, and completing less than 50% of the assignment is ineffective for learning.

We will not accept late assignments. Plan for this and aim to submit at least a day early. If you know you will have a problem submitting by the deadline, due to a personal issue that arises, please contact the instructor as early as possible to make a plan. If you have an emergency that prevents submission near the deadline, please contact the instructor right away. Retroactive reasons for delays are much harder to deal with in a fair way. If you submit a few minutes late (even up to an hour late), then this counts as being on time, to account for any small issues with uploading in eClass.

All assignments are individual. All the sources used for the problem solution must be acknowledged, e.g. web sites, books, research papers, personal communication with people, etc. Academic honesty is taken seriously; for detailed information see the University of Alberta Code of Student Behaviour.

Good luck!