Week 11 exercises

This is the eighth and last page of *assessed* questions, as described in the background notes. These questions form 70% of your mark for Week 11. The introductory questions in the notes and the Week 11 discussion group task form the remaining 30% of your mark for Week 11.

Unlike the questions in the notes, you'll not immediately see any example answers on this page. However, you can edit and resubmit your answers as many times as you like until the deadline (Friday 04 December 4pm UK time, UTC). This is a *hard deadline*: This course does not permit extensions and any work submitted after the deadline will receive a mark of zero. See the late work policy.

Queries: Please don't discuss/query the assessed questions on hypothesis until after the deadline. If you think there is a mistake in a question this week, please email Iain.

Please only answer what's asked. Markers will reward succinct to-the-point answers. You can put any other observations in the "Add any extra notes" button (but this is for your record, or to point out things that seemed strange, not to get extra credit). Some questions ask for discussion, and so are open-ended, and probably have no perfect answer. For these, stay within the stated word limits, and limit the amount of time you spend on them (they are a small part of your final mark).

Feedback: We'll return feedback on your submission via email by Friday 11 December.

Good Scholarly Practice: Please remember the University requirements for all assessed work for credit. Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (permitting access only to yourself). You may not publish your solutions after the deadline either.

1 The Laplace approximation

In note w10b, the second question was about inferring the parameter λ of a Poisson distribution based on an observed count r. You found the Laplace approximation to the posterior over λ given r.

Now reparameterize the model in terms of $\ell = \log \lambda$. After performing the change of variables[1], the improper prior on $\log \lambda$ becomes uniform, that is $p(\ell)$ is constant.

a) Find the Laplace approximation to the posterior over $\ell = \log \lambda$. Include the derivation of your result in your answer. [10 marks]

   [The website version of this note has a question here.]

b) Which version of the Laplace approximation is better? To help answer this question, we recommend that you plot the true and approximate posteriors of λ and ℓ for different values of the integer count r. However, we don't want your code or plots. Describe the relevant findings in no more than 150 words. [15 marks]

   [The website version of this note has a question here.]

2 Classification with unbalanced data

Classification tasks often involve rare outcomes, for example predicting click-through events, fraud detection, and disease screening. We'll restrict ourselves to binary classification, $y \in \{0, 1\}$, where the positive class is rare: $P(y = 1)$ is small.

[1] Some review of how probability densities work: Conservation of probability mass means that: $p(\ell)\mathrm{d}\ell = p(\lambda)\mathrm{d}\lambda$ for small corresponding elements $\mathrm{d}\ell$ and $\mathrm{d}\lambda$. Dividing and taking limits: $p(\ell) = p(\lambda)|\mathrm{d}\lambda/\mathrm{d}\ell|$, evaluated at $\lambda = \exp(\ell)$. The size of the derivative $|\mathrm{d}\lambda/\mathrm{d}\ell|$ is referred to as a Jacobian term.

We are likely to see *lots* of events before observing enough rare $y = 1$ events to train a good model. To save resources, it's common to only keep a small fraction f of the $y = 0$ 'negative examples'. A classifier trained naively on this sub-sampled data would predict positive labels more frequently than they actually occur. For Bayes classifiers we could set the class probabilities based on the original class counts (or domain knowledge). This question explores what to do for logistic regression.

We write that an input-output pair occurs in the world with probability $P(\mathbf{x}, y)$, and that the joint probability of an input-output pair $(\mathbf{x}, y)$ *and* keeping it (k) is $P(\mathbf{x}, y, k)$. Because conditional probabilities are proportional to joint probabilities, we can write:

$$P(y \,|\, \mathbf{x}, k) \propto P(\mathbf{x}, y \,|\, k) \propto P(\mathbf{x}, y, k) = \begin{cases} P(\mathbf{x}, y) & y = 1 \\ f\, P(\mathbf{x}, y) & y = 0. \end{cases} \tag{1}$$

A general result that you may quote in answering this question is that for any binary probability distribution:

$$P(z) \propto \begin{cases} c & z = 1 \\ d & z = 0, \end{cases} \tag{2}$$

we can write $P(z{=}1) = \sigma(\log c - \log d)$, where $\sigma(a) = 1/(1 + e^{-a})$.

a) We train a logistic regression classifier, with a bias weight, to match subsampled data, so that $P(y{=}1 \,|\, \mathbf{x}, k) \approx \sigma(\mathbf{w}^\top \mathbf{x} + b)$.

   i) Use the above results to argue that we should add $\log f$ to the bias parameter to get a model for the real-world distribution $P(y \,|\, \mathbf{x}) \propto P(y, \mathbf{x})$. [15 marks]

   [The website version of this note has a question here.]

   ii) Explain whether we have changed the bias in the direction that you would expect. Write no more than one sentence. [5 marks]

   [The website version of this note has a question here.]

b) Consider the following approach. We may wish to minimize the loss:

$$L = -\mathbb{E}_{P(\mathbf{x},y)}[\log P(y \,|\, \mathbf{x})]\,.$$

Multiplying the integrand by $1 {=} \frac{P(\mathbf{x}, \mathbf{y} \,|\, k)}{P(\mathbf{x}, \mathbf{y} \,|\, k)}$ changes nothing, so we can write:

$$\begin{aligned} L &= -\int \sum_y P(\mathbf{x}, y \,|\, k)\, \frac{P(\mathbf{x}, y)}{P(\mathbf{x}, y \,|\, k)}\, \log P(y \,|\, \mathbf{x})\, \mathrm{d}\mathbf{x} \\ &= -\mathbb{E}_{p(\mathbf{x},y \,|\, k)}\left[\frac{P(\mathbf{x}, y)}{P(\mathbf{x}, y \,|\, k)}\, \log P(y \,|\, \mathbf{x})\right] \\ &\approx -\frac{1}{N} \sum_{n=1}^{N} \frac{P(\mathbf{x}^{(n)}, y^{(n)})}{P(\mathbf{x}^{(n)}, y^{(n)} \,|\, k)}\, \log P(y^{(n)} \,|\, \mathbf{x}^{(n)}), \end{aligned}$$

where $\mathbf{x}^{(n)}, y^{(n)}$ come from the subsampled data. This manipulation is a special case of a trick known as *importance sampling* (see note w11a). We have converted an expectation under the original data distribution, into an expectation under the subsampling distribution. We then replaced the formal expectation with an average over subsampled data.

We can use the same idea to justify multiplying the gradients for $y{=}0$ examples by $1/f$, when training a logistic regression classifier on subsampled data.

$P(\mathbf{x}, y \,|\, k)$ was defined *up to a constant*. Thus the "importance weights" applied to each log probability are:

$$\frac{P(\mathbf{x}, y)}{P(\mathbf{x}, y \,|\, k)} \propto \begin{cases} \frac{P(\mathbf{x},y)}{P(\mathbf{x},y)} & y = 1 \\ \frac{P(\mathbf{x},y)}{f\,P(\mathbf{x},y)} & y = 0 \end{cases} \propto \begin{cases} 1 & y = 1 \\ 1/f & y = 0. \end{cases} \tag{3}$$

We can substitute these values of 1 or $1/f$ into the loss. The unknown constant scales the loss surface, but does not change where the minimum is.

The loss for a positive example is then $\log P(y\!=\!1 \,|\, \mathbf{x})$ as usual, whereas the loss for a negative example becomes $(1/f) \log P(y\!=\!0 \,|\, \mathbf{x})$. The gradients for negative examples are then also scaled by $(1/f)$.

This method says that if we throw away all but 1/1000 of the negative examples, we should up-weight the effect of each remaining negative example by 1000× to make up for it.

Compare this approach to the approach from part a) for training models based on subsampled data for binary classification, giving pros and cons of each. Write no more than 150 words. [25 marks]

[The website version of this note has a question here.]