Logistic regression with a latent binary variable and noisy labels

September 28, 2018

1 Model

Following [1], consider a set of N training data points $D = \{(\mathbf{x}_n, c_n) : 1 \leq n \leq N\}$ where $\mathbf{x}_n \in \mathbb{R}^M$ denote M-dimensional real-valued explanatory data (e.g. gene expression levels) and $c_n \in \{0, 1 \ldots K-1\}$ denotes a categorical label with K possible values (e.g. clinician-assigned label incorporating some degree of uncertainty).

We aim to fit this with a two-stage model, first regressing the explanatory data $\mathbf{x}_n$ to a latent binary variable representing the true underlying state $b_n \in \{0, 1\}$ (e.g. disease state), then modeling clinical labeling as a categorical variable $c_n|b_n$ that is conditionally-independent of the explanatory data given the true underlying state

$$\begin{aligned} P(b=1|\mathbf{x}, \mathbf{u}) &= \sigma\left(\mathbf{u}^T \mathbf{x}\right) \\ P(c=k|b=j, \mathbf{z}) &= z_{j,k} \end{aligned}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic function, $\mathbf{u} \in \mathbb{R}^M$ are parameters for the logistic regression model, and $\mathbf{z}$ are probability parameters for the label observation model.

We put a Laplace double-exponential (Lasso) prior on $\mathbf{u}$, and a flattish[1] Dirichlet prior on each row of $\mathbf{z}$

$$\begin{aligned} P(\mathbf{u}) &\propto \prod_{m=1}^{M} \exp(-|u^{(m)}|) \\ P(\mathbf{z}|\alpha) &\propto \prod_{j \in \{0,1\}} \prod_{k=0}^{K-1} z_{j,k}^{\alpha_{j,k}} \end{aligned}$$

subject to the constraint $\sum_{k=0}^{K-1} z_{j,k} = 1 \; \forall \; j \in \{0, 1\}$, i.e. the parameters $z_{j,k}$ for each value of j must lie on the $(K-1)$-dimensional simplex.

[1] For identifiability of b, we need to break the symmetry of the Dirichlet prior slightly; e.g. by setting $\alpha_{b,c} = 1$ for (b, c) pairs that "agree", and $\alpha_{b,c} = 0$ for all other (b, c) pairs.

This is equivalent to Section 2.2 of [1], with the sum over j in equation (8) of that paper constrained to $j \in \{0, 1\}$ instead of $j \in \{0, 1 \ldots K-1\}$. The paper derives a conjugate gradient optimization algorithm, and proves its convergence.

1.1 Quartile approach

An alternate model is to use the interpretation of logistic regression where a latent *continuous-valued* random variable (obtained by adding logistically-distributed noise to $\mathbf{u}^T\mathbf{x}$) is used to obtain the labels (b, c), e.g. with c corresponding to the quartiles.

I haven't pursued this model, as the assumption that c corresponds to quartiles of the latent variable underlying logistic regression seems like a possible misfit to the situation of arbitrarily designated clinical labels (although, conceivably, my assumption that c is independent of $\mathbf{x}$ given b is just as bad, or worse).

2 EM algorithm

How to use the training data D to fit the logistic model weights $\mathbf{u}$ and label error probabilities $\mathbf{z}$? One approach is to use the EM (Expectation Maximization) algorithm [2], treating the binary-valued latent variables $B = \{b_n\}$ as *missing data*, the dataset $D = (X, C)$ as *observed data* (with inputs $X = \{\mathbf{x}_n\}$ and observed labels $C = \{c_n\}$), and $\theta = (\mathbf{u}, \mathbf{z})$ as the *parameters* to be fit by the algorithm.

The conjugate gradient parameter optimization approach derived by [1] may well be superior to the EM method. However I've outlined the EM approach here for reference.

The joint likelihood including observed and missing data is

$$\begin{aligned} P(B, C, \theta|X) &= P(\theta)P(B, C|\theta, X) \\ &= P(\mathbf{u})P(\mathbf{z}|\alpha)P(B|\mathbf{u}, X)P(C|\mathbf{z}, B) \\ &= P(\mathbf{u})P(\mathbf{z}|\alpha)\prod_{n=1}^{N} P(b_n|\mathbf{u}, \mathbf{x}_n)P(c_n|\mathbf{z}, b_n) \end{aligned}$$

The marginal likelihood to be maximized, using observed data only, is

$$\begin{aligned} P(C, \theta|X) &= \sum_B P(B, C, \theta|X) \\ &= P(\mathbf{u})P(\mathbf{z}|\alpha)\prod_{n=1}^{N}\sum_{j\in\{0,1\}} P(b_n = j|\mathbf{u}, \mathbf{x}_n)P(c_n|\mathbf{z}, b_n = j) \end{aligned}$$

At the $(i+1)$'th iteration, the parameters found by the EM algorithm are given by maximizing the expected log-likelihood

$$
\begin{aligned}
\theta^{(i+1)} &= \operatorname{argmax}_\theta \; \mathcal{E}\left(\theta || \theta^{(i)}\right) \\
\mathcal{E}\left(\theta || \theta^{(i)}\right) &= \sum_B P(B|\theta^{(i)}, X, C) \log P(B, C, \theta | X) \\
&= \log P(\mathbf{u}) + \log P(\mathbf{z}|\alpha) + \sum_B P(B|\theta^{(i)}, X, C) \left[\log P(B|\mathbf{u}, X) + \log P(C|\mathbf{z}, B)\right] \\
&= \log P(\mathbf{u}) + \log P(\mathbf{z}|\alpha) \\
&\quad + \sum_n \sum_{j \in \{0,1\}} P(b_n = j|\theta^{(i)}, \mathbf{x}_n, c_n) \left[\log P(b_n = j|\mathbf{u}, \mathbf{x}_n) + \log P(c_n|\mathbf{z}, b_n = j)\right] \\
&= \mathcal{E}_\mathbf{u} \; + \; \mathcal{E}_\mathbf{z} \\
\mathcal{E}_\mathbf{u} &= \log P(\mathbf{u}) + \sum_n \left[(1 - \beta_n^{(i)}) \log(1 - \sigma\left(\mathbf{u}^T \mathbf{x}_n\right)) + \beta_n^{(i)} \log \sigma\left(\mathbf{u}^T \mathbf{x}_n\right)\right] \\
\mathcal{E}_\mathbf{z} &= \log P(\mathbf{z}|\alpha) + \sum_n \left[(1 - \beta_n^{(i)}) \log z_{0,c_n} + \beta_n^{(i)} \log z_{1,c_n}\right] \\
\beta_n^{(i)} &= P(b_n = 1|\theta^{(i)}, \mathbf{x}_n, c_n) \\
P(b_n = 1|\theta, \mathbf{x}_n, c_n) &= \frac{1}{1 + \frac{P(c_n, b_n = 0|\theta, \mathbf{x}_n)}{P(c_n, b_n = 1|\theta, \mathbf{x}_n)}} \\
&= \frac{1}{1 + \frac{(1 - \sigma(\mathbf{u}^T \mathbf{x}_n)) z_{0,c_n}}{\sigma(\mathbf{u}^T \mathbf{x}_n) z_{1,c_n}}}
\end{aligned}
$$

The maximization of $\mathcal{E}_\mathbf{u}$ w.r.t. $\mathbf{u}$ is a weighted, Lasso-penalized logistic regression (the weights being the $\beta_n^{(i)}$). A suggested implementation of this maximization is outlined in Section 2.1.

The maximization of $\mathcal{E}_\mathbf{z}$ w.r.t. $\mathbf{z}$ can be performed using Lagrange multipliers.

$$
\begin{aligned}
\mathcal{E}_\mathbf{z} &= \sum_{j,k} \gamma_{0,k} \log z_{j,k} \\
\gamma_{0,k} &= \alpha_{0,k} + \sum_{n:\; c_n = k} \left(1 - \beta_n^{(i)}\right) \\
\gamma_{1,k} &= \alpha_{1,k} + \sum_{n:\; c_n = k} \beta_n^{(i)}
\end{aligned}
$$

To maximize $\mathcal{E}_\mathbf{z}$ subject to the constraint that $\sum_{k=0}^{K-1} z_{j,k} = 1$ for each $j \in \{0, 1\}$, we introduce Lagrange multipliers λ_j and maximize the following instead

$$
\mathcal{L}(\mathbf{z}; \lambda_0, \lambda_1) = \mathcal{E}_\mathbf{z} + \sum_j \lambda_j \left(1 - \sum_k z_{j,k}\right)
$$

The maximum occurs where $\frac{d\mathcal{L}}{dz_{j,k}} = 0$ and $\frac{d\mathcal{L}}{d\lambda_j} = 0$, which leads to

$$
\operatorname{argmax}_{z_{b,c}} \mathcal{E}_\mathbf{z} = \frac{\gamma_{b,c}}{\sum_k \gamma_{b,k}}
$$

2.1 Implementation of weighted logistic regression in R

The maximization of $\mathcal{E}_{\mathbf{u}}$ w.r.t. $\mathbf{u}$ can be performed using R's `glmnet()` function (generalized linear model regression with Lasso prior) with `family = binomial(link = "logit")` (logistic regression is equivalent to binomial-family GLM regression with the "logit" link function).

R's GLM-fitter allows *weighting* of the training examples; that is, an augmented dataset $D' = \{(\mathbf{x}'_n, b'_n, f'_n) : 1 \leq n \leq N'\}$ where f'_n is a weight (by default 1), loosely equivalent (when integer-valued) to the number of times datapoint $(\mathbf{x}_n, b_n)$ was observed, or its *frequency*.

To implement the M-step in the $(i+1)$'th iteration of EM, we construct a weighted pseudo-dataset D' containing $N' = 2N$ weighted training examples (that is, two for every example in the original unweighted dataset D). The first N are labeled as negatives, the remaining N as positives; the weights are set using $\beta_n^{(i)}$, the posterior probability inferences from the previous step of EM. Specifically, for $1 \leq n \leq N$

$$
\begin{aligned}
\mathbf{x}'_n &= \mathbf{x}_n \\
\mathbf{x}'_{N+n} &= \mathbf{x}_n \\
b'_n &= 0 \\
b'_{N+n} &= 1 \\
f'_n &= 1 - \beta_n^{(i)} \\
f'_{N+n} &= \beta_n^{(i)}
\end{aligned}
$$

The weights can be supplied to the R code using using the `weights` argument to `glmnet()`.

The weighted logistic regression fit can then be worked into an R program that implements EM. Pseudocode for this algorithm is as follows

- Set $\theta^{(1)}$ to some "sensible" initial values
- For $i \in \{1, 2, 3 \ldots\}$ do:
  - Calculate $\beta_n^{(i)}$ using current $\theta^{(i)} = (\mathbf{u}^{(i)}, \mathbf{z}^{(i)})$
  - Set $\mathbf{z}^{(i+1)} \leftarrow \mathrm{argmax}_{\mathbf{z}}(\mathcal{E}_{\mathbf{z}})$ by counting & normalizing
  - Set $\mathbf{u}^{(i+1)} \leftarrow \mathrm{argmax}_{\mathbf{u}}(\mathcal{E}_{\mathbf{u}})$ using `glmnet()` with `weights`
- Loop over i continues while $\log \frac{P(C, \theta^{(i+1)}|X)}{P(C, \theta^{(i)}|X)} > \epsilon$ for some EM convergence threshold ϵ

References

[1] Jakramate Bootkrajang and Ata Kabán. Label-noise robust logistic regression and its applications. In *Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part I*, ECML PKDD'12, pages 143–158, Berlin, Heidelberg, 2012. Springer-Verlag.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.