COMPUTO

ISSN 2824-7795

# A Bayesian hierarchical model of categorical data rating and classification

Bob Carpenter  Center for Computational Mathematics, Flatiron Institute

**Abstract**

We introduce a Bayesian model of categorical data rating (aka coding or annotation) and classification that reparameterizes the model of Dawid and Skene (1979) on the log odds scale in order to add item-level effects and hierarchical, multivariate priors. Rater effects capture raters' accuracy and bias and item-level effects capture the bias introduced by the items being classified such as difficulty. We show that item-level effects are crucial for ensuring calibrated predictions. We use multivariate priors to capture mean task accuracy, bias, and correlation among responses, which allows sharper and better calibrated predictions observed or new raters, as found in an ongoing data rating task with crowdsourcing. We extend the analysis of Raykar et al. (2010), using item-level predictors (aka features or covariates) to jointly estimate (aka train) a classifier. With. no item-level predictors, our model reduces to the standard prevalence-only model. Moving beyond the point estimates of Raykar et al., we train a fully Bayesian classifier and show how proper posterior predictive inference outperforms plug-in point estimates. We further demonstrate that training on a probabilistic corpus, either sequentially by weighting or jointly with full Bayes, leads to implicit regularization and outperforms alternative non-probabilistic approaches to training.

*Keywords:* crowdsourcing, data rating, Bayesian modeling, multivariate priors, classification, item difficulty

## Contents

# 1 Crowdsourcing classifiers

Supervised training of classifiers is based on labeled data sets. Labeled data typically arises from humans or systems rating the data (also known as "coding" or "annotating" in different literatures). Because human and machine raters are never 100% accurate, the problem arises as to how to deal with disagreements in ratings. For example, given a radiology image, one human rater might say it shows a stage 1 cancer tumor and another may say it is stage 2 or not a tumor at all; or given a social media post, one rater might say it is positive toward a product and another might say it is neutral or perhaps not even about the product. How do we adjudicate these disagreements among raters and get on with building classifiers? One traditional approach is to vote—use multiple raters and take a majority vote. Often this is done in stages, where if two raters disagree, a third is brought in to settle the dispute. This can be problematic when both raters make the same error or when there is disparity in accuracy or correlated bias among the raters. And in the end, these voting schemes leave us with no measure of certainty in our ratings. Another traditional approach is to censor data where there is disagreement—that is, use only items in the training data on which all of the raters agreed. This approach may lead to clean data, but it will not be representative of the "wild type" data from which it was selected.

In order to make the best use of our data, we need to appropriately model it to correct for the bias and accuracy of the raters as well as the difficulty and biases introduced by the items being annotated (see, e.g., (Dawid and Skene 1979; Passonneau and Carpenter 2014)). In the end, we will be left with a "soft" data set, with probabilities assigned to outcomes for each item. A traditional data set uses a one-hot encoding (where one category has probabity 1 and the others probability 0) and is often derived by assigning the most probable category. We will show in this paper that taking the best category is inferior to selecting a category at random based on the probability distribution, which in turn is inferior to training directly with the probabilistic weights.

Among the contributions of this paper are a new crowdsourcing and classifier training model that introduces (1) item-level effects for difficulty and item bias, (2) multivariate priors on item-level and rater-level effects, and (3) joint classifier and data set training with full Bayesian inference. The model strictly generalizes the widely used models of Dawid and Skene (1979) and (**raykar2010larning?**) and extends inference from point estimation to full Bayes. We show how (1) item-level effects are necessary to achieve calibrated predictions, (2) the multivariate priors borrow strength to improve inference, especially for raters with no or little data, and (3) training a model jointly with full Bayesian inference leads to better inference than a factorized approach.

# 2 Probabilistic training of probabilistic classifiers

Our goal is to train a probabilistic classifier, by which we mean one which returns probabilistic predictions. Ideally, such predictions would be calibrated in the usual statistical sense of having appropriate coverage (see, e.g., Gneiting, Balabdaoui, and Raftery (2007)). Roughly speaking, calibration means that if the system says there's an 80% chance of an item being a category,

then there really is an 80% chance of it being that category. For example, with $N$ predictions of an 80% chance of rain, the actual number of rain days will be distributed as binomial$(N, 0.8)$.

Even though we are building probabilistic classifiers, our training data is typically assigned a unique category in gold-standard training sets. When we crowdsource our data, we have the opportunity to develop probabilistic training data sets. That is, we might have some disagreement among data raters and some inherent noise in their ratings, so that we are not 100% sure of any of the categories in our data set. For example, with probabilistic training we might train a somewhat ambiguous handwritten image as being 90% likely to be the digit '6', 9% likely to be the digit '0', and 1% likely to be the digit '9'.

We show that taking training data uncertainty into account leads to tighter estimates as measured by proper scoring metrics such as squared error or log loss (see, e.g., Gneiting and Raftery (2007)). Specifically, we provide simulations that show why probabilistic training is preferable, and barring that, why sampling a single label from the probability distribution for the training data is preferable to deterministically selecting the most probable category.

## 2.1 Logistic regression classifiers

We will evaluate a binary logistic regression classifier with a vector of predictors for each item. We choose logistic regression because it is simple—the same argument holds for more expressive classifiers such as neural networks. Suppose we have $L$ predictors and $N$ data items, so that the predictors form a matrix $x \in \mathbb{R}^{N \times L}$. The observed outcomes are binary, so take the form of a boolean vector $z \in \{0, 1\}^N$.

### 2.1.1 Data generating model for logistic regression

The data generating model for the true category $z_n \in \{0, 1\}$ for item $n \in 1{:}N$ is based on an intercept parameter $\alpha \in \mathbb{R}$ and slope vector $\beta \in \mathbb{R}^L$,

$$z_n \sim \text{bernoulli}\Big(\text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big)\,,$$

where $x_n \in \mathbb{R}^L$ is the $n$-th row of the data matrix $x$.

### 2.1.2 Weakly informative priors for regression coefficients

We will assign independent, weakly informative priors to the intercept

$$\alpha \sim \text{normal}(0, 3)$$

and to the slopes

$$\beta_l \sim \text{normal}(0, 3),$$

for $l \in 1{:}L$. These are chosen to be in line with standardized predictors (i.e., columns of $x$ are approximately mean zero and unit variance) for logistic regression (see, e.g., (Gelman et al. 2008)). For example, $\text{logit}^{-1}(-6) = 0.0025$ and $\text{logit}^{-1}(6) = 0.9975$, and we don't expect effects (intercept or slope times predictor) to be larger than this in our classifiers.

### 2.1.3 Posterior predictive inference

Given training data $x, y$ and parameters $\alpha$ and $\beta$, Bayes's rule allows us to express the posterior distribution over the parameters up to a proportion as

$$\begin{aligned} p(\alpha, \beta \mid x, y) &\propto p(\alpha, \beta) \cdot p(y \mid x, \alpha, \beta) \\ &= \text{normal}(\alpha \mid 0, 3) \cdot \textstyle\prod_{l=1}^{L} \text{normal}(\beta_l \mid 0, 3) \\ &\quad \cdot \textstyle\prod_{n=1}^{N} \text{bernoulli}\big(y_n \,\big|\, \text{logit}^{-1}\big(\alpha + \beta \cdot x_n^\top\big)\big). \end{aligned}$$

Now suppose we have a new predictor $\tilde{x} \in \mathbb{R}^L$. The posterior predictive distribution over the associated outcome $\tilde{y} \in \{0, 1\}$ will be

$$\begin{aligned} p(\tilde{y} \mid \tilde{x}, y, x) &= \mathbb{E}[\, p(\tilde{y} \mid \tilde{x}, \alpha, \beta) \mid x, y] \\ &= \int_{\mathbb{R}^{1+L}} p(\tilde{y} \mid \tilde{x}, \alpha, \beta) \cdot p(\alpha, \beta \mid x, y) \, \mathrm{d}(\alpha, \beta). \end{aligned}$$

### 2.1.4 Inference with point estimates

We also consider estimation with point estimates, where instead of full Bayes, we take a point estimate of our parameters and then plug in for inference. That is, we set

$$\alpha^*, \beta^* = \arg\max_{\alpha, \beta} \; p(y \mid \alpha, \beta, x) \cdot p(\alpha, \beta),$$

which can be viewed as a penalized maximum likelihood estimate (MLE) if the prior $p(\alpha, \beta)$ is considered a penalty, or viewed as a maximum a posteriori (MAP) estimate. Inference for new items then proceeds by plugging in parameters,

$$p(\tilde{y} \mid \tilde{x}, x, y) \approx p(\tilde{y} \mid \tilde{x}, \alpha^*, \beta^*),$$

where the point estimates $\hat{\alpha}, \hat{\beta}$ are derived as MAP estimates as above.

Another form of point estimate that we may use is the Bayesian posterior mean. For example, our estimate for $\alpha$ is

$$\begin{aligned} \hat{\alpha} &= \mathbb{E}[\, \alpha \mid x, y] \\ &= \int_{\mathbb{R}} \alpha \cdot p(\alpha \mid x, y) \, \mathrm{d}\alpha, \end{aligned}$$

and $\hat{\beta}$ is defined the same way as $\mathbb{E}[\beta \mid x, y]$. Such an estimate is similar to what would result from taking a point estimate from a variational approximation to the posterior and plugging in the point estimate for inference.

## 2.2 Simulated data

### 2.2.1 Simulated predictors

We will sample two situations, one where the predictors are independent and one where they're highly correlated. In the independent case, we generate predictor vectors according to a standard normal distribution,

$$x_n \sim \text{normal}(0, \mathrm{I}).$$

In the correlated case, we will generate

$$x_n \sim \text{normal}(0, \Sigma),$$

where $\Sigma$ is a symmetric, positive-definite, $L \times L$ matrix with entries

$$\Sigma_{i,j} = \rho^{|i-j|}$$

for some $\rho \in (-1, 1)$. Both approaches to produce predictors that are marginally standard normal (i.e., $x_{n,l} \sim \text{normal}(0, 1)$).

### 2.2.2 Simulated regression coefficients

We will simulate regression coefficients independently according to a slightly more limited distribution than our priors,

$$\alpha \in \text{normal}(0, 1),$$

and

$$\beta \in \text{normal}(0, 2 \cdot \text{I}).$$

### 2.2.3 Simulated outcomes

For a given item $n \in 1 : N$ with a predictor (row) vector $x_n \in \mathbb{R}^L$, we have

$$\Pr[y_n = 1] = \text{logit}^{-1}(\alpha + x_n \cdot \beta).$$

We consider three different ways to train.

1. *Sampling*: We begin with the approach that follows the actual generative model. In this approach, we probabilistically sample the category of each item according to its probability,

$$y_n \sim \text{bernoulli}\Big(\text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big).$$

2. *Best category*: In this approach, which is popular in machine learning settings, we take the most probable category for each training instance,

$$y_n = \text{I}\left(\text{logit}^{-1}(\alpha + x_n \cdot \beta) > \frac{1}{2}\right),$$

   where $\text{I}(c)$ is the indicator function, with $\text{I}(c) = 1$ if the condition $c$ is true and $\text{I}(c) = 0$ otherwise. In words, $y_n = 1$ if its probability of being 1 is greater than one half and $y_n = 0$ if the probability is one half or lower.

3. *Weights*: In this approach, we never generate a true $y_n$, but instead just store the weights,

$$w_n = \text{logit}^{-1}(\alpha + x_n \cdot \beta).$$

   In this sceneario, we do not need to resolve the true value of $y_n$ for training.

## 2.3 Probababilistic training as linear regression on log odds

In approach (1) and (2), where we sample or take the best guess for $y_n$, we train our logistic regression as usual. In approach (3) with weights, there are two options. In one approach, we can train a linear regression of $\text{logit}(w_n)$ on $x_n$. This method exploits the relationship between a logistic regression and a linear regression on the log odds. While this is possible for logistic regression, it's not always an option in more general classifiers, so we consider one more approach to weighted training.

## 2.4 Probabilistic training as weighted regularization

In the second approach to weighted training, we train a logistic regression with weighted instances. The first approach is supported by standard software, but the second requires weighting of the training instances. This is easy to carry out in a probabilistic programming framework, where we can define the target likelihood using weight as:

$$p(w \mid x, \alpha, \beta) = \prod_{n=1}^{N} \text{bernoulli}\left(1 \,\middle|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\right)^{w_n} \cdot \text{bernoulli}\left(0 \,\middle|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\right)^{1-w_n},$$

or on the log scale as we will implement the density,

$$\log p(w \mid x, \alpha, \beta) = \sum_{n=1}^{N} w_n \cdot \log \text{bernoulli}\Big(1 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big) + (1-w_n) \cdot \log \text{bernoulli}\Big(0 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big).$$

That is, we train with both the positive and negative example category ($y_n = 1$ *and* $y_n = 0$), but with different weights ($w_n$ and $1 - w_n$). If we evaluate the gradient of the likelihood, the contributions for $y_n = 1$ and $y_n = 0$ point in opposite directions,

$$\nabla_{\alpha,\beta} \log \text{bernoulli}\Big(1 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big) = -\nabla_{\alpha,\beta} \log \text{bernoulli}\Big(0 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big),$$

so that we wind up shrinking the update as $\Pr[y_n = 1] \to \frac{1}{2}$. The total gradient for a weighted item is

$$\begin{aligned}
&\nabla_{\alpha,\beta}\, w_n \cdot \log \text{bernoulli}\Big(1 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big) + (1 - w_n) \cdot \log \text{bernoulli}\Big(0 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big) \\
&\quad = (2 \cdot w_n - 1) \cdot \nabla_{\alpha,\beta} \log \text{bernoulli}\Big(1 \,\Big|\, \text{logit}^{-1}(\alpha + x_n \cdot \beta)\Big).
\end{aligned}$$

That is, we train the positive category weighted by $2 \cdot w_n - 1$. The overall effect is to scale the contribution of examples based on their uncertainty. Looking at the boundaries, if $w_n = 1$ or $w_n = 0$, we reduce to ordinary training with weight 1 or -1 (equivalently, ordinary training with $y_n = 1$ or $y_n = 0$). When $w_n < 1$ and $w_n > 0$, then we have the effect of downweighting the effect on training the most likely category. For a completely uncertain example, where $\Pr[y_n = 1] = \frac{1}{2}$, the overall gradient $\nabla_{\alpha,\beta}\, p(y_n \mid \alpha, \beta) = 0$ and the training example has no effect.

### 2.5 Simulation results

## 3 Observed rating data

In this section, we describe the observed data that we model as well as the constants required.

### 3.1 Rating data

We assume there are $K \in \mathbb{N}$ categories into which items are classified, $I \in \mathbb{N}$ items being rated, and $J \in \mathbb{N}$ raters. We assume there are $N \in \mathbb{N}$ ratings, with $y_n \in 1{:}K$ being the rating given by rater $jj[n]$ for item $ii[n]$. The result is a long-form table of $N$ rows; the first few rows of an example are shown in Table 1.

Table 1: Long-form data format for ratings. Annotation $n$ is for item $ii[n]$ by annotator $jj[n]$, who supplied rating $y[n]$. For example, rating $n = 3$ was made for item $ii[3] = 1$ by rater $jj[3] = 6$, who provided label $y[3] = 6$. Three raters, with ids 1, 2, and 6, rated item $i = 1$ providing labels 4, 4, and 3 respectively.

| n | ii | jj | y |
|---|---|---|---|
| 1 | 1 | 1 | 4 |
| 2 | 1 | 2 | 4 |
| 3 | 1 | 6 | 3 |
| 4 | 2 | 3 | 1 |
| 5 | 2 | 4 | 1 |
| 6 | 3 | 2 | 6 |
| ⋮ | ⋮ | ⋮ | ⋮ |

This data format is flexible enough to allow each item to be rated by a zero or more raters. While it is possible to represent a single rater rating the same item multiple times, our models will treat the ratings as independent.

### 3.2 Item-level predictors

In addition to the ratings, we will assume there are $L$ predictors (features, etc.) for each item. We let $x \in \mathbb{R}^{I \times L}$ be the data matrix, with rows $x_i \in \mathbb{R}^L$ being the $L$-vector of predictors for item $i \in 1{:}I$. We model intercepts separately and thus do not assume that there is a column of 1s in the matrix $x$. Although it would be possible to have rater-level predictors, such as the geographical location or age or sex of the rater, we do not consider that extension in this paper.

## 4 Data-generating process

We formulate our statistical model generatively in the sense that it is able to generate a complete data set given the items and predictors. We will start with the model for the items then consider

a model for the ratings.

## 4.1 Prevalence and classification

We will assume that each item $i \in 1:I$ has a true category $z_i \in 1:K$. The $z[i]$ are not observed and may be considered missing data and represented by means of a discrete parameter in the model. We model the category based on item-level predictors using a logistic regression, where we assume $\beta \in \mathbb{R}^{L \times K}$ is our matrix of regression coefficients and $\alpha \in \mathbb{R}^K$ is an intercept.

$$z_i \sim \text{categorical}\big(\text{softmax}(\alpha + \beta \cdot x_i^\top)\big)\,,$$

where $x_i$ is the $i$-th row of the matrix $x$ and $\text{softmax}(u) = \exp(u)/\text{sum}(\exp(u)) \in \Delta^{K-1}$, with exp() applied elementwise. We will be able to use the fitted model to make predictions for new items not in the training set assuming we have their predictor vectors. That is, the result will be a classifier for new items.

In the case where we have no item-level predictors (i.e., $L = 0$), our model reduces to an intercept-only model where $\text{softmax}(\alpha) \in \Delta^{K-1}$ represents the simple prevalence of the categorical outcomes.

## 4.2 Generative model for ratings

We will model the sampling distribution for rating using a logistic regression with effects for the item being rated and the rater performing the rating. This section describes the four types of effects we assume on that rating. All of the effects are vector parameters in $\mathbb{R}^K$ (i.e., the size of the number of categories).

### 4.2.1 Intercept for global bias

In order to model potential biases in ratings that are independent of the category of the item being rated, we will assume there is an intercept term in our logistic regression, $\xi \in \mathbb{R}^K$. A high value for $\xi_k$ means there is an overall bias toward category $k$ whereas a low value represents an overall bias away from category $k$.

### 4.2.2 True category effect

The category assigned to item $i$ by a rater is strongly influenced by the true category $z_i$ of that item. We thus assume there is an effect $\psi_k$ based on the true category $k$ of the item being rated. This will contribute a term $\psi_{z_{ii[n]}}$ for the $n$-th rating, which has category $ii[n]$ and true category $z_{ii[n]}$. Another way to consider the true category effect is as the prior location for the item effects for an item of category $k$ and as a prior location for rater responses to items of true category $k$.

### 4.2.3 Rater effects

In order to allow raters to vary in their accuracies and biases, we will model each rater to have their own probabilistic response to items of a given true category. Specifically, we assume that

each rater $j \in 1:J$ has a response simplex $\theta_{j,k} \in \Delta^{K-1}$ which says how they respond to items of category $k$, all else being equal. A perfect rater has $\theta_{j,k,k'}$ equal to 1 if $k = k'$ and 0 otherwise. That is, $\theta_{j,k,k}$ represents rater $j$'s accuracy on items of category $k$ and the off-diagonal elements of $\theta_j$ represent the biases.

### 4.2.4 Item effects

We will further assume that each item $i \in 1:I$ has a vector of effects $\varphi_i \in \mathbb{R}^K$. If $\varphi_i = 0$, the item has no effect on ratings and raters will just return results according to $\theta_{j,k}$ for items of category $k$. If $\varphi_{i,z[i]}$ is high, the item is relatively easy to rate, whereas if it's low, the item is difficult to rate, with the other terms determining the response bias.

### 4.2.5 Sampling distribution for ratings

Given the rating and item-level effects, the generative model for ratings is a multi-logit regression,

$$y_n \sim \text{categorical}\Big(\text{softmax}\Big(\xi + \psi_{z_{ii[n]}} + \varphi_{ii[n]} + \theta_{jj[n],\, z_{ii[n]}}\Big)\Big).$$

Breaking this down, item $ii[n] \in 1:I$ is being given a rating of $y_n \in 1:K$ by rater $jj[n] \in 1:J$. The true rating for item $ii[n]$ is $z_{ii[n]} \in 1:K$. The effects being added are vectors in $\mathbb{R}^K$. The softmax() function transforms the unconstrained vector to a simplex, which means the vector components are on the log probability scale. The first effect is the intercept $\xi$, which accounts for overall bias in response by the raters. The second term $\psi_{z_{ii[n]}}$ is the effect of the true category $zii[n]$. The third term $\varphi_{ii[n]}$ is the effect of the item being rated. The final term $\theta_{jj[n],\, z_{ii[n]}}$ is the effect of rater $jj[n]$ responding to an item whose true category is $z_{ii[n]}$.

## 4.3 Full data likelihood and marginal likelihood

For each item $i \in 1:I$ being rated, there is a latent discrete parameter $z_i \in 1:K$ for its true category, zero or more ratings, and an item-level coefficient $\varphi_i \in \mathbb{R}^K$. We will marginalize out the $z_i$ explicitly and the value of $\varphi$ through sampling. To simplify subsequent notation, we define

$$\text{idx}(i, ii) = \{n \,:\, ii[n] = i\},$$

the set of indexes of ratings for item $i \in 1:I$ and raters $ii[n] \in 1:I$ for $n \in 1:N$.

The so-called full data likelihood is $p(y, z \mid x, \omega)$, where $y$ is the observed ratings and $z$ is the latent true categories, conditioned on the item-level predictor matrix $x$ and the full sequence of parameters $\omega = \alpha, \beta, \xi, \psi, \phi, \theta$. We can derive the likelihood $p(y \mid x, \omega)$ by marginalizing out the $z$, first noting that

$$p(y, z \mid x, \omega) = \prod_{i=1}^{I} p\big(y[\text{idx}(i, ii)], z[i] \,\big|\, x, \omega\big),$$

because the coverage of the indexes is exhaustive and we make a single selection of $z[i]$ per item. We then expand the data-item specific latent category $z[i]$ in he usual way by summation,

$$p(y[\text{idx}(i, ii) \,\big|\, x, \omega) = \sum_{k=1}^{K} p(y[\text{idx}(i, ii)], z[i] = k \,\big|\, x, \omega).$$

Then we use the chain rule to decompose

$$\begin{aligned} p\big(y[\text{idx}(i,ii)], z[i] = k \,\big|\, x, \omega\big) &= p\big(z[i] = k \,\big|\, \alpha, \beta, x\big) \cdot p(y[\text{idx}(i,ii)] \,|\, z[i] = k, \xi, \psi, \varphi, \theta) \\ &= p\big(z[i] = k \,\big|\, \alpha, \beta, x\big) \cdot \textstyle\prod_{n \in \text{idx}(i,ii)} p\big(y[n] \,\big|\, z[i] = k, \xi, \psi, \varphi, \theta\big) \, . \end{aligned}$$

# 5 Priors

## 5.1 Prior for prevalence regression coefficients

For the prevalence regression, we provide weakly informative priors for the components of the intercept $\alpha \in \mathbb{R}^K$,

$$\alpha_k \sim \text{normal}(0, 3),$$

and the components of the slopes $\beta \in \mathbb{R}^{L \times K}$,

$$\beta_{l,k} \sim \text{normal}(0, 3).$$

## 5.2 Sum-to-zero constraint for unconstrained simplex parameters

The underlying dimensionality of a simplex is one less than the number of categories it ranges over. In order to match our unconstrained parameterization on the log odds scale to the dimensionality of a simplex, we will constrain it to sum to zero. This removes what would otherwise be an additive non-identifiability in the regression that would allow us to add a constant $c$ to every dimension any of the effects without changing the sampling distribution.

## 5.3 Prior for global effect

We assign the global intercept a weakly informative prior,

$$\xi \sim \text{normal}(0, 3 \cdot \text{I}),$$

where I is the identity matrix.

## 5.4 Prior for category-level effects

Without any prior knowledge of which categories are likely to be correlated with category $k$, we will assume a weakly inforamtive prior on the category-level effects,

$$\psi_k \sim \text{normal}(0, 3 \cdot \text{I}),$$

for $k \in 1 : K$.

### 5.5 Prior for item-level effects

The item-level effects $\varphi_i \in \mathbb{R}^K$ are assigned multivariate normal priors centered at zero,

$$\varphi_i \sim \text{normal}(0, \Sigma^{\varphi}),$$

for $i \in 1:I$, where $\Sigma^{\varphi}$ is a symmetric, positive-definite covariance matrix parameteter.

### 5.6 Prior for rater-level effects

The rater-level effects $\theta_{j,k} \in \mathbb{R}^K$ are assigned to a prior conditioned on the true category $k$,

$$\theta_{j,k} \sim \text{normal}(0, \Sigma^{\theta}_k),$$

where $\Sigma^{\theta}_k$ is a positive definite covariance matrix for $k \in 1:K$.

### 5.7 Hyperpriors for the covariance of varying effects

We have symmetric, positive-definite covariance parameters $\Sigma^{\varphi}$ and $\Sigma^{\theta}_k$ for $k \in 1:K$. We factor covariance matrices into a vector of scales and a correlation matrix,

$$\Sigma^{\varphi} xtrmdiag(\sigma^{\varphi}) \cdot \Omega^{\varphi} \cdot \text{diag}(\sigma^{\varphi}),$$

for strictly positive $\sigma^{\varphi} \in \mathbb{R}^K_+$ and a correlation matrix $\Omega^{\varphi}$ (i.e., a symmetric, positive definite matrix with unit diagonal). We factor the rater-level covariances by category $k$, taking

$$\Sigma^{\theta}_k = \text{diag}(\sigma^{\theta}_k) \cdot \Omega^{\theta}_k \cdot \text{diag}(\sigma^{\theta}_k)$$

for $k \in 1:K$, where $\sigma^{\theta}_k \in \mathbb{R}^K_+$ is a vector of strictly positive scales and $\Omega^{\theta}_k$ is a correlation matrix.

The components of the scale parameters are assigned weakly informative half-normal priors independently by component, taking

$$\sigma^{\theta}_{k,k'}, \sigma^{\varphi}_k \sim \text{normal}_+(0, 3),$$

for $k, k' \in 1:K$. We assign Lewandowski-Kurowicka-Joe (LKJ) priors to the correlation matrix,

$$\Omega^{\theta}_k, \Omega^{\varphi}_k \sim \text{LKJ}(5),$$

for $k \in 1:K$, where the LKJ density is defined for a symmetric positive-definite, unit-diagonal correlation matrix $\Omega$ and shape $\eta > 0$ by

$$\text{LKJ}(\Omega \mid \eta) \propto \det(\Omega)^{\eta - 1}.$$

For $\eta = 1$, the LKJ distribution is uniform over correlation matrices $\Omega$. For $\eta > 1$, it concentrates mass around the unit correlation matrix (with $\eta < 0$ it concentrates toward the boundaries). Thus when used as a prior on a correlation matrix parameter, it has the effect of shrinking the correlation estimates (i.e., the off-diagonal elements of an estimated $\Omega$).

# 6 Previous work

Our model of categorical data rating is very similar to models used in epidemiology for the results of diagnostic tests with unknown sensitivity and specificity. Raters play the role of diagnostic tests with items being classified playing the role of patients. Our model directly derives from the model of Dawid and Skene (1979), which appeared in the epidemiology literature. Following Passonneau and Carpenter (2014), the rating model presented in this paper reparameterizes Dawid and Skene's model on the log odds scale to allow multiple additive effects such as item-level effects and introduces priors for regularization. It extends the model of Passonneau and Carpenter by adding multivariate priors with covariance parameters in order to capture dependencies between categorical responses across items and annotators.

If the item-level effects are all zero, the likelihood reduces to a reparameterized version of Dawid and Skene's. Following Raykar et al. (2010), the model introduced here also jointly specifies a logistic regression classifier for item prevalence; this is a common move in epidemiology to model the dependence of disease prevalence on predictors such as time, location, age, sex, etc.

Following Paun et al. (2018), we use full Bayesian inference rather than the point estimation calculated by expecation maximization (EM) as employed by Dawid and Skene, Raykar et al., and Passonneau and Carpenter. Full Bayes takes parameter estimation uncertainty into account for posterior predictive inference, and typically leads to better calibrated posterior predictive inference. We further show that jointly training a logistic regression classifier leads to better performance, which we trace to the regularization effects provided by probabilistic training.

# Session information

```
{r session-info} sessionInfo()
```


Dawid, Alexander Philip, and Allan M Skene. 1979. "Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 28 (1): 20–28.

Gelman, Andrew, Aleks Jakulin, Maria Grazia Pittau, and Yu-Sung Su. 2008. "A Weakly Informative Default Prior Distribution for Logistic and Other Regression Models." *The Annals of Applied Statistics* 2 (4): 1360–83.

Gneiting, Tilmann, Fadoua Balabdaoui, and Adrian E Raftery. 2007. "Probabilistic Forecasts, Calibration and Sharpness." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69 (2): 243–68.

Gneiting, Tilmann, and Adrian E Raftery. 2007. "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association* 102 (477): 359–78.

Passonneau, Rebecca J, and Bob Carpenter. 2014. "The Benefits of a Model of Annotation." *Transactions of the Association for Computational Linguistics* 2: 311–26.

Paun, Silviu, Bob Carpenter, Jon Chamberlain, Dirk Hovy, Udo Kruschwitz, and Massimo Poesio. 2018. "Comparing Bayesian Models of Annotation." *Transactions of the Association for Computational Linguistics* 6: 571–85.

Raykar, Vikas C, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca

Bogoni, and Linda Moy. 2010. "Learning from Crowds." *Journal of Machine Learning Research* 11 (4).