1 Opening

Nuggeteer gives scores for each response-nugget pair, and the judgement files give a human judgement as to whether each response contains each nugget. We construct Bayesian models that are descriptive of the scores, resposnes, and nuggets, but generative with respect to the judgements. We model judgement assignments as being caused directly by the scores, according to a noisy threshold function. In other words, if the score exceeds some threshold, the response will be judged to contain the nugget, and if it does not, it will not, except for probabilities of error. To put it symbolically, if we parametrize the noisy threshold function by some threshold α and low and high error rates (meaning, error rates below and above the threshold value, respectively) ϵ_l, ϵ_h, referred to collectively by θ, the probability of a "yes" judgement as a function of the nuggeteer score s will be given by:

$$p_y(s|\theta) = \begin{cases} 1 - \epsilon_h & s > \alpha \\ \epsilon_l & \text{otherwise} \end{cases}$$

and of "no", conversely, by:

$$p_n(s|\theta) = 1 - p_y(s|\theta).$$

Then, for any given score-judgement pair $d = (s, j)$,

$$p(d|\theta) = \begin{cases} p_y(s|\theta) & j = \text{“yes”} \\ p_n(s|\theta) & j = \text{“no”} \end{cases}$$

and, for any given set D of score-judgement pairs d_i,

$$p(D|\theta) = \prod_i p(d_i|\theta).$$

Note that this produces a probability distribution over the judgements given the scores, responses, nuggets, and parameters.

2 Models

Now that we have a theory of scores, errors, and thresholds, several reasonable possibilities for specific models present themselves. They vary thus:

- Are the high and low error rates different?
- Is there one global threshold, or should the thresholds vary by year, question, or nugget?
- Do the error rates (be they equal or different) vary by year, question, or nugget?

We can consider models that answer these questions in every combination of ways.

The worth of a Bayesian model M is the posterior distribution

$$p(\theta|D, M) = \frac{p(D|\theta)p(\theta|M)}{p(D|M)}$$

that it induces over its parameters θ, and the derived inference machinery, which gives, for a putative data point d,

$$p(d|D, M) = \int_\theta p(d|\theta)p(\theta|D, M).$$

In general practice, as well as in our particular case, the likelihood, $p(D|\theta)$, is part of the definition (as was, indeed, presented at the end of Section **??**), and is easily computable; the prior, $p(\theta|M)$, is dispensed with by making it uniform;[1] and it is the normalization constant, $p(D|M)$, that presents a problem. If one wishes only to compare probabilities within a single model, it is sometimes possible to avoid dealing with the normalization constant, but in our case, especially since we have multiple models, we will deal with it. As it happens, dealing with the normalization constant is remarkably similar to inference, so the effort we are about to undertake will not be wasted.

As to the process of computing said normalization constants, let us examine, as a basic case, the model M_b stating: "Thresholds and error rates are global, and error rates are not constrained to be equal." What, then, is $p(D|M_b)$, for a given collection D of response-nugget pairs and their scores?

$$p(D|M_b) = \int_\theta p(D|M_b, \theta)p(\theta|M_b) \quad (1)$$

[1]The error probabilities, being probabilities, naturally range from zero to one, and Nuggeteer only outputs scores in the range from zero to one, so that is a natural bound on the thresholds as well.

$$
\begin{aligned}
&= \int_0^1 \int_0^1 \int_0^1 p(D|M_b, \epsilon_l, \epsilon_h, \alpha) p(\epsilon_l, \epsilon_h, \alpha|M_b) d\alpha d\epsilon_h d\epsilon_l && (2)\\
&= \int_0^1 \int_0^1 \int_0^1 p(D|M_b, \epsilon_l, \epsilon_h, \alpha) d\alpha d\epsilon_h d\epsilon_l && (3)\\
&= \int_0^1 \int_0^1 \int_0^1 (1-\epsilon_l)^{c_1(\alpha)} \epsilon_l^{c_2(\alpha)} \epsilon_h^{c_3(\alpha)} (1-\epsilon_h)^{c_4(\alpha)} d\alpha d\epsilon_h d\epsilon_l && (4)\\
&= \int_0^1 \int_0^1 \sum_i (1-\epsilon_l)^{c_1(\alpha)} \epsilon_l^{c_2(\alpha)} \epsilon_h^{c_3(\alpha)} (1-\epsilon_h)^{c_4(\alpha)} (s_{i+1} - s_i) d\epsilon_h d\epsilon_l && (5)
\end{aligned}
$$

Where **??** and **??** follow by defintion, **??** follows by assumption of uniform prior, **??** follows, with definitions of the c's as counts of appropriate events, from the functional form of $p(D|M_b, \theta)$, and **??** follows by sorting the data points by increasing score. Here we define the c's as the following counts of data points:

$$
\begin{aligned}
c_1(\alpha) &= |\{d_i = (s_i, j_i) | s_i \leq \alpha, j_i = \text{“no”}\}|, \\
c_2(\alpha) &= |\{d_i = (s_i, j_i) | s_i \leq \alpha, j_i = \text{“yes”}\}|, \\
c_3(\alpha) &= |\{d_i = (s_i, j_i) | s_i > \alpha, j_i = \text{“no”}\}|, \\
c_4(\alpha) &= |\{d_i = (s_i, j_i) | s_i > \alpha, j_i = \text{“yes”}\}|.
\end{aligned}
$$

Observe that the c's are exhaustive, and so, for any fixed α, $c_1(\alpha) + c_2(\alpha) + c_3(\alpha) + c_4(\alpha) = |D|$. Though the integral and sum **??** may seem daunting for large c_*, it (or, rather, its logarithm, since it is so small) are, in fact, possible to compute with reasonable precision and in reasonable time. How to do this is the subject of Appendix **??**.

Now consider any grouping G of the data points into disjoint, exhaustive groups $g \subseteq D$. Any reasonable such grouping (e.g. group by question or group by nugget) leads to two more reasonable models: a model M_G^θ which asserts that each group g has its own (independent) set of parameters $\theta_g = (\epsilon_{lg}, \epsilon_{hg}, \alpha_g)$, and a model M_G^α which asserts that each group g has its own (independent) threshold α_g, but the error rates ϵ_l, ϵ_h are global. (We take the model M_G^ϵ that asserts that the error rates are local per group but the threshold is global not to be reasonable).

If the grouing G divides D into one group that contains all the data points in D, then $M_G^\theta = M_G^\alpha = M_b$, as detailed above. Otherwise, some math bears doing to find the relationships among M_G^θ, M_G^α, and M_b. Let the number of groups $|G|$ be n, let the counts $c(\alpha_g)$ count only the data points in a given

group g, and let us begin with M_G^θ.

$$p(D|M_G^\theta) = \int_{\theta_1}\int_{\theta_2}\cdots\int_{\theta_n}\prod_{i=1}^{n}(1-\epsilon_{lg_i})^{c_1(\alpha_{g_i})}\epsilon_{lg_i}^{c_2(\alpha_{g_i})}\epsilon_{hg_i}^{c_3(\alpha_{g_i})}(1-\epsilon_{hg_i})^{c_4(\alpha_{g_i})}.$$

While this integral looks absolutely awful, we are fortunate in that as θ_{g_i} varies, neither ϵ_{*g_j} nor $c_*(\alpha_{g_j})$ vary for any $j \neq i$. Therefore, the integral of the product becomes the product of the integrals, and

$$p(D|M_G^\theta) = \prod_{i=1}^{n}\int_{\theta_{g_i}}(1-\epsilon_{lg_i})^{c_1(\alpha_{g_i})}\epsilon_{lg_i}^{c_2(\alpha_{g_i})}\epsilon_{hg_i}^{c_3(\alpha_{g_i})}(1-\epsilon_{hg_i})^{c_4(\alpha_{g_i})},$$

which is just a product of models like M_b, but on separate data sets g:

$$p(D|M_G^\theta) = \prod_{i=1}^{n} p(g_i|M_b).$$

Now to the other model, M_G^α. It is quite similar, except that there are only two ϵ variables. If we integrate with respect to them on the furthest outside, we find

$$p(D|M_G^\alpha) = \int_{\epsilon}\int_{\alpha_{g_1}}\int_{\alpha_{g_2}}\cdots\int_{\alpha_{g_n}}\prod_{i=0}^{n}(1-\epsilon_l)^{c_1(\alpha_{g_i})}\epsilon_l^{c_2(\alpha_{g_i})}\epsilon_h^{c_3(\alpha_{g_i})}(1-\epsilon_h)^{c_4(\alpha_{g_i})}.$$

Now, by the same observation of non-variance,

$$p(D|M_G^\alpha) = \int_{\epsilon}\prod_{i=0}^{n}\int_{\alpha_{g_i}}(1-\epsilon_l)^{c_1(\alpha_{g_i})}\epsilon_l^{c_2(\alpha_{g_i})}\epsilon_h^{c_3(\alpha_{g_i})}(1-\epsilon_h)^{c_4(\alpha_{g_i})}.$$

This integral is not the same as **??**, but can still be approxiamted numerically. A natural numerical integration method amounts to putting a discrete prior on the ϵ parameters. In our case, we work with giving probability $\frac{1}{100}$ to each value of ϵ in $\left\{\frac{k}{100}\right\}$.

3 Inference

Given a new TREC system for evaluation, it is desirable to compute an F-measure for it, as an estimate of how it would score in the actual TREC competition, and assignments of the individual nuggets to the system's response, as justification and development information.

In the Bayesian setting, it is more convenient to compute probabilities of nugget assignments, and then compute an expected F-measure from them. A particular model M produces a posterior on parameters $p(\theta_M|M)$. Suppose we are given a set of responses $R = \{r_i\}$ to a given question. Suppose the question has nuggets $N = \{n_j\}$. Then the probability $p(n_j|R, M)$ of the response set R containing the nugget n_j is given in terms of the probabilities of each response r_i containing n_j in the natural way,

$$p(n_j|R, M) = 1 - \prod_i (1 - p(n_j|r_i, M)).$$

The $p(n_j|r_i, M)$ are derived from the Nuggeteer scores we are modeling directly from the assumptions in Section **??**. For any pair (r_i, n_j), Nuggeteer produces a score s_{ij}. For a given set of parameters θ_M, we have

$$p(n_j|r_i, \theta_M) = p_y(s_{ij}|\theta_M),$$

so for the whole trained model we have

$$p(n_j|r_i, M) = \int_{\theta_M} p_y(s_{ij}|\theta_M) p(\theta_M|M). \quad (6)$$

The computations involved in (**??**) are the subject of Appendix **??**.

With the probabilities $p(n_j|R, M)$ in hand, we can turn to the final task of computing expected F-measure. Unfortunately, we face the snag that F-measure contains precision- and recall-dependent terms in the denominator, and expectations do not divide. Fortunately, we can work around this well enough. Since F-measure is indifferent, modulo the vital/okay distinction, to *which* nuggets are present in the response, it suffices to compute from the nugget probabilities probability distributions on the counts of vital and okay nuggets present in the response. From there, we need only evaluate the definitions of expectation and F-measure to produce expected F.

A Computation

So, now that we have all these beautiful integrals, products, and summations, how should we go about actually computing them? Let us begin with the problem, both more important and more readily solvable, of integrating with respect to α. Fortunately, since the values of the $c(\alpha)$ only change at a finite number of values of α, this integral reduces to a summation. Unfortunately,

the values being summed are products of large powers of numbers less than 1, or perhaps integrals of such products, and as such can be too small for standard floating point arithmetic. Fortunately, we have their logarithms, and can carry out logspace summation upon them, as described in the next paragraph.

Suppose we have some x_i, and want to know $\sum x_i$, but the x_i are too small for standard floating-point arithmetic. Let $a_i = \ln x_i$. Then

$$\sum x_i = \sum e^{a_i}.$$

Now, for any $A = \ln X$,

$$\ln \sum x_i = \ln \left(X \sum \frac{x_i}{X} \right) = \ln X + \ln \sum \frac{x_i}{X} = A + \ln \sum e^{a_i - A}.$$

If we choose $A = \max(a_i)$, the largest of the $e^{a_i - A}$ will be 1, so any others that remain too small for floating point can safely be ignored, as they will not contribute meaningfully to the sum.

Let us now turn to the problem of integrating with respect to ϵ. In the M_G^α models, there are just two ϵ parameters and many α_g parameters. If we choose to integrate the ϵ's outside, there will be no closed form (known to the authors) for the final integrals, but two parameters are not too hard to integrate numerically.

In the M_G^θ models, on the other hand, there are many ϵ parameters, but only one α parameter per pair of ϵ parameters. In this case, it is useful to integrate with respect to ϵ on the inside, for then they are independent of each other, and ***FIND REFERENCE*** tells us that

$$\int x^n (1-x)^m dx = \frac{x^{n+1} {}_2F_1(n+1, -m; n+2; x)}{n+1} + C,$$

where ${}_2F_1$ is the hypergeometric function. Unfortunately, computing the value of ${}_2F_1$ for arguments as large as $c(\alpha)$ is beyond the capabilities of the GSL (***REFERENCE?***). Fortunately, ***REFERENCE*** provides a helpful formula:

$${}_2F_1(a, b; c; 1) = \frac{\Gamma(c)\Gamma(c-a-b)}{\Gamma(c-a)\Gamma(c-b)},$$

so, in our case,

$$\int_0^1 \epsilon^n (1-\epsilon)^m d\epsilon \;\; = \;\; \frac{1^{n+1} {}_2F_1(n+1, -m; n+2; 1)}{n+1} - 0$$

$$
\begin{aligned}
&= \frac{1}{n+1}\frac{\Gamma(n+2)\Gamma(m+1)}{\Gamma(1)\Gamma(m+n+2)} \\
&= \frac{(n+1)!(m)!}{(n+1)(n+m+1)!},
\end{aligned}
$$

the logarithm of which is readily computable.

B Inference Details

Given a system's responses to a question and the associated nugget-score pairs, we wish to compute the probabilities that the system's responses will contain the nuggets, and from those probabilities, we can compute the expected F-measure for that system on that question. As discussed in Section **??**, that amounts to computing, for each system response r_i and nugget n_j, with corresponding score s_{ij},

$$
p(n_j|r_i, M) = \int_{\theta_M} p_y(s_{ij}|\theta_M)p(\theta_M|M). \tag{7}
$$

In the particular case of a model of the M_G^α class, with the uniform priors we have been assuming, the parameters are two global ϵ paramters, ϵ_l and ϵ_h, and one parameter α_g for each group in the grouping G. If we let g be the group of the nugget n_j, the other $\alpha_{g'}$ parameters are irrelevant and thus integrate out to 1, and the integral (**??**) reduces to

$$
\begin{aligned}
p(n_j|r_i, M_G^\alpha) &= \int_0^1\int_0^1\int_0^1 p(n_j|r_i, \alpha_g, \epsilon_l, \epsilon_h)p(\alpha_g, \epsilon_l, \epsilon_h|M_G^\alpha)d\alpha_g d\epsilon_h d\epsilon_l \qquad (8)\\
&= \int_0^1\int_0^1 \left(\int_0^1 p(n_j|r_i, \alpha_g, \epsilon_l, \epsilon_h)p(\alpha_g|\epsilon_l, \epsilon_h, M_G^\alpha)d\alpha_g\right) p(\epsilon_l, \epsilon_h|M_G^\alpha)d\epsilon_h d\epsilon_l \qquad (9)
\end{aligned}
$$

Let us consider for a moment the case when the $\{\epsilon\}$ are fixed. Then, because of the structure of p_y,

$$
\begin{aligned}
p(n_j|r_i, \epsilon_l, \epsilon_h, M_G^\alpha) &= \int_0^1 p(n_j|r_i, \alpha_g, \epsilon_l, \epsilon_h)p(\alpha_g|\epsilon_l, \epsilon_h, M_G^\alpha)d\alpha_g \\
&= \int_0^1 p_y(s_{ij}|\alpha_g, \epsilon_l, \epsilon_h)p(\alpha_g|\epsilon_l, \epsilon_h, M_G^\alpha)d\alpha_g \\
&= \epsilon_h\int_0^{s_{ij}} p(\alpha_g|\epsilon_l, \epsilon_h, M_G^\alpha)d\alpha_g + \epsilon_l\int_{s_{ij}}^1 p(\alpha_g|\epsilon_l, \epsilon_h, M_G^\alpha)d\alpha_g.
\end{aligned}
$$

For fixed $\{\epsilon\}$ the structure of M_G^α entails a piecewise constant posterior on α_g, with changes in the constant at relevant scores in the training data. Therefore, the latter two integrals reduce to sums and can be computed without any further mathematical insight.[2]

Unfortunately, the $\{\epsilon\}$ parameters are more difficult to deal with precisely. While the posterior on $\{\epsilon\}$ can be written down,

$$\begin{aligned} p(\epsilon_l, \epsilon_h | D, M_G^\alpha) &= \frac{p(D|\epsilon_l, \epsilon_h, M_G^\alpha)(p(\epsilon_l, \epsilon_h | M_G^\alpha) = 1)}{p(D|M_G^\alpha)} \\ &= \frac{\prod_{g\in G} \int_0^1 p(D|\epsilon_l, \epsilon_h, \alpha_g) p(\alpha_g | M_G^\alpha) d\alpha_g}{p(D|M_G^\alpha)}, \\ p(D|M_G^\alpha) &= \int_0^1 \int_0^1 \left(\prod_{g\in G} \int_0^1 p(D|\epsilon_l, \epsilon_h, \alpha_g) p(\alpha_g | M_G^\alpha) d\alpha_g \right) d\epsilon_l d\epsilon_h, \end{aligned}$$

neither these integrals nor their substitution into (**??**) are amenable to symbolic evaluation. We chose, therefore, to replace the continuous uniform prior on $\{\epsilon\}$ with a discrete uniform one,[3] converting the outer integrals both above and in (**??**) into sums (and using the piecewise constance of the posterior on α_g to evaluate the inner one). This has much the same effect as estimating said integrals with a sample-and-add numerical integration technique, but has the benefit of sampling consistently across all the integrations, and having an interpretation in the language of the model.

---

[2]For the terminally curious, they are computed thus: Order said relevant scores s_g by increasing value as s_g^k, and let k^* be the index such that $s_g^{k^*} < s_{ij} < s_g^{k^*+1}$. Then, for fixed $\{\epsilon\}$,

$$\begin{aligned} p(n_j | r_i, \epsilon_l, \epsilon_h, M_G^\alpha) &= \sum_{k<k^*} \epsilon_h (s_g^{k+1} - s_g^k) p\left(\alpha_g = \frac{s_g^k + s_g^{k+1}}{2} | \epsilon_l, \epsilon_h, M_G^\alpha\right) + \\ &\quad + \sum_{k>k^*} \epsilon_l (s_g^{k+1} - s_g^k) p\left(\alpha_g = \frac{s_g^k + s_g^{k+1}}{2} | \epsilon_l, \epsilon_h, M_G^\alpha\right) + \\ &\quad + \left(\epsilon_h (s_{ij} - s_g^{k^*}) + \epsilon_l (s_g^{k^*+1} - s_{ij})\right) p\left(\alpha_g = \frac{s_g^{k^*} + s_g^{k^*+1}}{2} | \epsilon_l, \epsilon_h, M_G^\alpha\right). \end{aligned}$$

Computing this sum involves nothing more than a finite number of evaluations of $p(\alpha_g | \epsilon_l, \epsilon_h, M_G^\alpha)$ and a finite number of arithmetical operations.

[3]In this instance, ϵ_l ranging by hundredths from 0.01 to 0.2, and ϵ_h from 0.01 to 0.35.