The Annals of Applied Statistics
2015, Vol. 9, No. 3, 1394–1414
DOI: 10.1214/15-AOAS831

DISTRIBUTED MULTINOMIAL REGRESSION

BY MATT TADDY

University of Chicago

This article introduces a model-based approach to distributed computing for multinomial logistic (softmax) regression. We treat counts for each response category as independent Poisson regressions via plug-in estimates for fixed effects shared across categories. The work is driven by the high-dimensional-response multinomial models that are used in analysis of a large number of random counts. Our motivating applications are in text analysis, where documents are tokenized and the token counts are modeled as arising from a multinomial dependent upon document attributes. We estimate such models for a publicly available data set of reviews from Yelp, with text regressed onto a large set of explanatory variables (user, business, and rating information). The fitted models serve as a basis for exploring the connection between words and variables of interest, for reducing dimension into supervised factor scores, and for prediction. We argue that the approach herein provides an attractive option for social scientists and other text analysts who wish to bring familiar regression tools to bear on text data.

1. Introduction. This article is motivated by data sets that include *counts* in a massive number of *categories*, such as text corpora (counts for words), browser logs (counts on websites), and website tracking (counts of clicks). The unit upon which counts are observed—for example, a "document" for text or a "user" in web analysis—is annotated with *attributes*, additional information about each document (author, date, etc.) or user (age, purchases, etc.). Much of contemporary Big Data analysis involves some exploration, inference, and prediction of, or controlling for, the relationship between these attributes and the associated very high-dimensional counts.

Say $\mathbf{c}_i$ is a vector of counts in d categories, summing to $m_i = \sum_j c_{ij}$, accompanied by a p-dimensional attribute vector $\mathbf{v}_i$ on observation unit i of n total. For example, in the archetypal text mining application, $\mathbf{c}_i$ are counts for words in document i annotated with metadata $\mathbf{v}_i$. We connect attributes and counts through a big multinomial logistic regression model,

$$\mathrm{p}(\mathbf{c}_i|\mathbf{v}_i, m_i) = \mathrm{MN}(\mathbf{c}_i; \mathbf{q}_i, m_i)$$
$$\text{where } q_{ij} = \frac{e^{\eta_{ij}}}{\Lambda_i},\ \eta_{ij} = \alpha_j + \mathbf{v}_i'\boldsymbol{\varphi}_j \quad \text{and} \quad \Lambda_i = \sum_{k=1}^{d} e^{\eta_{ik}}. \tag{1}$$

Received December 2013; revised April 2015.

Key words and phrases. Distributed computing, MapReduce, logistic regression, lasso, text analysis, multinomial inverse regression, computational social science.

The multinomial denoted MN here has, for unit i, category j, probability q_{ij}, and size m_i. This model can be computationally expensive to estimate for a large number of response categories (i.e., big $\mathbf{c}_i$ dimension d). Even a single likelihood evaluation is costly, due to the sum required for each normalizing *intensity* $\Lambda_i = \sum_{k=1}^{d} e^{\eta_{ik}}$. The methodological innovation of the current article is to replace Λ_i with initial estimates, then condition upon these plug-ins when estimating (1) through d individual Poisson regressions for counts in each category j. This model-based factorization allows one to *partition* computation across many independent machines, so with enough processors the system of (1) is fit in the time required for a single Poisson regression.

We refer to this framework as *distributed multinomial regression*, or DMR. Our work here extends ideas from Taddy (2013a), which introduced the strategy of *multinomial inverse regression* (MNIR). That article argues for estimation of models like (1) as the first step in an inverse regression routine for predicting elements of new $\mathbf{v}_i$. However, Taddy (2013a) relies upon a fitting algorithm that collapses response counts across equal $\mathbf{v}_i$, and hence scales *only* for a small number of attributes (i.e., when p is just one or two). That article is also focused exclusively on applications in attribute prediction. The purpose of the current article is thus twofold: to supply techniques for estimation when both $\mathbf{c}$ and $\mathbf{v}$ are high dimensional, and to motivate how these models can be useful in many aspects of analysis and inference.

Much of the paper is devoted to an example analysis of reviews on Yelp—an Internet platform for feedback on various establishments, including restaurants, barbers, schools, and much else. This data set has a rich feature set associated with a wide variety of reviews. The data are also publicly available, after (free) registration on the data mining contest website kaggle.com. Moreover, our technology is provided in the `distrom` package for `R` and Yelp analysis code is cataloged at github.com/mataddy/yelp. Public access is essential here: our goal is to provide a complete template for analysis of high-dimensional count data.

The estimation strategy is detailed in Section 2, including model factorization, plug-ins for Λ_i, and regularization path estimation within each parallel regression. Methods are illustrated in the short classification example of Section 3, which shows utility for DMR not only in big d but also as a speedup for small d multinomial regressions. Finally, Section 4 runs through our full Yelp application, detailing model estimation and a variety of analyses. These analyses each correspond to a different inferential goal.

Exploration: *What words are associated with funny or useful content?* Here, we interpret the fitted regression model at the level of word loadings. We emphasize that these loadings represent *partial effects*—connections between text and attributes that arise after controlling for collinearity between attributes—and we describe how the interpretations change when controlling for more or less confounding information.

Dimension reduction: *Which reviews have the most funny or useful content?* Once the model has been fit, it acts as a map between text and attributes. We describe how to use this map to obtain *sufficient reductions*: low-dimensional scores that summarize text content *directly* relevant to a given attribute. We show that the sufficient reduction for, say, funny votes, is a seemingly better judge of humor than the original yelp voters (because reviews can generate funny votes for correlated but unfunny reasons).

Prediction: *What will be the usefulness or hilarity of a new review?* We consider performance of the above sufficient reductions as input to prediction algorithms, and find that they can match or outperform comparable techniques that use the full text as input.

Treatment effects: *Does user tenure lead to higher ratings?* Finally, we show how to use the sufficient reductions as synthetic controls in regressions that wish to remove from a targeted treatment effect the influence of any correlated text content. That is, the sufficient reductions act like a text-based propensity; in our application, we use them to see if older users are more positive in their ratings even if we control for the change in their review content.

Section 5 closes with a discussion and some practical advice for applications in text mining. We argue that, for many social science and business applications, the methods herein make regression of text onto observed attributes an attractive alternative to techniques such as topic modeling which require estimation and interpretation of a latent space.

2. Methods: Estimation in distribution. We adopt terminology from text analysis for the remainder and refer to each unit i as a "document" and each category j as a "word."[1] Suppose that every document–word count c_{ij} has been drawn independently $\text{Po}(e^{\eta_{ij}})$—Poisson with intensity (i.e., mean) $e^{\eta_{ij}}$. The joint document likelihood for $\mathbf{c}_i$ then factorizes as the product of a multinomial distribution for individual counts conditional on total count m_i and a Poisson distribution on m_i:

$$\text{p}(\mathbf{c}_i) = \prod_j \text{Po}\big(c_{ij}; e^{\eta_{ij}}\big) = \text{MN}(\mathbf{c}_i; \mathbf{q}_i, m_i)\,\text{Po}(m_i; \Lambda_i). \tag{2}$$

This well-known result has long been used by statisticians to justify ignoring whether sampling was conditional on margin totals in analysis of contingency tables. Birch (1963) showed that the maximum likelihood estimate (MLE) of $\mathbf{q}_i$ is unchanged under a variety of sampling models for 3-way tables *under the constraint* that $\Lambda_i = m_i$. This is satisfied at the MLE for a saturated model. Palmgren

[1]Even in text mining this is a simplification; each j could be a combination of words or any other language token.

(1981) extends the theory to log-linear regression with $\eta_{ij} = \alpha_j + \mu_i + \boldsymbol{\varphi}_j' \mathbf{v}_i$, showing that the Fisher information on coefficients is the same regardless of whether or not you have conditioned on m_i so long as μ_i in the Poisson model is estimated at its conditional MLE,

$$\mu_i^\star = \log\left(\frac{m_i}{\sum_j e^{\eta_{ij}}}\right). \tag{3}$$

Most commonly, (2) is invoked when applying multinomial logistic regression: totals m_i are then ancillary and the μ_i drop out of the likelihood. Our DMR framework takes the opposite view: if we are willing to fix estimates $\hat{\mu}_i$ potentially not at their MLE (we will argue for $\hat{\mu}_i = \log m_i$), then the factorized Poisson likelihood can be analyzed independently across response categories.[2] As highlighted in the Introduction, this yields distributed computing algorithms for estimation on previously impossible scales. Indeed, we have observed in text and web analysis a recent migration from multinomial models—say, for latent factorization—to Poisson model schemes; see Gopalan, Hofman and Blei (2013) as an example. From the perspective of this article, such strategies are Big Data approximations to their multinomial precursors.

2.1. *Estimating baseline intensity.* The negative log likelihood implied by (1) is proportional to

$$\sum_{i=1}^{n}\left[m_i \log\left(\sum_{j=1}^{d} e^{\eta_{ij}}\right) - \mathbf{c}_i' \boldsymbol{\eta}_i\right]. \tag{4}$$

It is easy to verify that adding observation fixed effects μ_i to each η_{ij} in (4) leaves the likelihood unchanged. In contrast, the corresponding Poisson model, unconditional on m_i, has negative log likelihood proportional to

$$\sum_{j=1}^{d}\sum_{i=1}^{n}\left[e^{\mu_i+\eta_{ij}} - c_{ij}(\mu_i + \eta_{ij})\right] \tag{5}$$

with gradient on each μ_i of $g(\mu_i) = e^{\mu_i}\sum_j e^{\eta_{ij}} - m_i$, and is clearly sensitive to these observation "baseline intensities." As mentioned above, the solution for the parameters of η_{ij} is unchanged between (4) and (5) if each μ_i is set to its conditional MLE in (3).

Unfortunately, if our goal is to *separate* inference for $\boldsymbol{\varphi}_j$ across different j, the MLE formula of (3) will create a computational bottleneck: each category-j Poisson regression requires updates to $\boldsymbol{\mu}^\star = [\mu_1^\star \cdots \mu_n^\star]'$ during estimation. Distributed

[2]In an older version of this idea, Hodges and Le Cam (1960) introduce a Poisson approximation to the binomial distribution, for which McDonald (1980) provides error bounds and extension to multinomials.

computation precludes such communication, and we instead use the simple plug-in estimator

$$\hat{\mu}_i = \log m_i. \tag{6}$$

This choice is justified as optimal in a few simple models, and we rely upon empirical evidence to claim it performs well in more complex settings.[3]

The gradient of the Poisson likelihood in (5) on μ_i at our plug-in is $g(\hat{\mu}_i) = m_i(\sum_i e^{\eta_{ij}} - 1)$. Define the plug-in MLEs $\hat{\boldsymbol{\eta}}_i = [\hat{\eta}_{i1} \cdots \hat{\eta}_{id}]'$ as those which minimize the Poisson objective in (5) under $\mu_i = \hat{\mu}_i$. Then in the three simple settings below, $g(\hat{\mu}_i) = 0$ for $\boldsymbol{\eta}_i = \hat{\boldsymbol{\eta}}_i$. This implies that $\hat{\mu}_i$ is the optimal joint MLE estimate for μ_i, and thus that $\{\hat{\boldsymbol{\eta}}_i, \hat{\mu}_i\}$ minimize the Poisson objective in (5) while $\{\hat{\boldsymbol{\eta}}_i\}$ minimizes the logistic multinomial objective in (4):

- In a saturated model, with each η_{ij} free, $\hat{\eta}_{ij} = \log(c_{ij}) - \hat{\mu}_i = \log(c_{ij}/m_i)$ and $g(\hat{\mu}_i) = 0$.
- With intercept-only $\eta_{ij} = \alpha_j$, the Poisson MLE is $\hat{\alpha}_j = \log \sum_i c_{ij} - \log \sum_i e^{\hat{\mu}_i} = \log(\sum_i c_{ij}/M)$ where $M = \sum_i m_i$, and $g(\hat{\mu}_i) = m_i(\sum_j \sum_i c_{ij}/M - 1) = 0$.
- Consider a single $v_i \in \{0, 1\}$ such that $\eta_{ij} = \alpha_j + v_i\varphi_j$. Write $C_{vj} = \sum_{i:v_i=v} c_{ij}$ and $M_v = \sum_{i:v_i=v} m_i = \sum_j C_{vj}$. Then the Poisson MLE are $\hat{\alpha}_j = \log(C_{0j}/M_0)$ and $\hat{\varphi}_j = \log(C_{1j}/M_1) - \log(C_{0j}/M_0)$, so that $g(\hat{\mu}_i) = m_i(\sum_j C_{v_i j}/M_{v_i} - 1) = 0$.

Of course, these examples do not form a general result: the situation is more complicated with correlated covariates or under regularization. But they illustrate analytically why we might expect the performance we have seen empirically: estimates based upon $\hat{\mu}_i = \log m_i$ do not suffer in out-of-sample validation. The resulting benefit is huge, as using a plug-in allows estimation of the Poisson regression equations to proceed in complete isolation from each other. See the Appendix for an example MapReduce [Dean and Ghemawat (2008)] implementation.

2.2. *Parallel Poisson regressions.* Given baseline intensities fixed as $\hat{\mu}_i = \log m_i$, each of our d separate Poisson regressions has negative log likelihood proportional to

$$l(\alpha_j, \boldsymbol{\varphi}_j) = \sum_{i=1}^{n} [m_i e^{\alpha_j + \mathbf{v}_i'\boldsymbol{\varphi}_j} - c_{ij}(\alpha_j + \mathbf{v}_i'\boldsymbol{\varphi}_j)]. \tag{7}$$

You are free to use your favorite estimation technique for each parallel regression. This section outlines our specific approach: "gamma lasso" L_1 regularized deviance minimization.

[3]Note that, when compared to (3), the plug-in replaces $\sum_j e^{\alpha_j + \mathbf{v}_i'\boldsymbol{\varphi}_j}$ with 1. Adding a constant to each α_j leaves probabilities unchanged, so this can be made to hold without affecting fit.

In high-dimensional regression, it can be useful to regularize estimation through a penalty on coefficient size. This helps to avoid over-fit and stabilize estimation. A very common form of regularization imposes L_1 coefficient costs [i.e., the lasso of Tibshirani (1996)], which, due to a nondifferentiable cost spike at the origin, yields variable selection: some coefficient estimates will be set to exactly zero. Our results here use *weighted L_1 regularization*

$$(8)\quad \hat{\alpha}_j, \hat{\boldsymbol{\varphi}}_j = \underset{\alpha_j, \boldsymbol{\varphi}_j}{\arg\min} \left\{ l(\alpha_j, \boldsymbol{\varphi}_j) + n\lambda \sum_{k=1}^{p} \omega_{jk} |\varphi_{jk}| \right\} \qquad \text{where } \lambda, \omega_{jk} \geq 0.$$

Penalty size λ acts as a *squelch* that determines what you measure as signal and what you discard as noise. In practice, since optimal λ is unknown, one solves a *regularization path* of candidate models minimizing (8) along the grid $\lambda_1 > \lambda_2 > \cdots > \lambda_T$. Inference is completed through selection along this path, with optimal λ_t chosen to minimize cross-validation (CV) or information criteria (IC; e.g., Akaike's AIC) estimated out-of-sample (OOS) deviance (i.e., to minimize the average error for a given training algorithm when used to predict new data). Crucially, *selection is applied independently for each category j regression*, so that only a single set of coefficients need be communicated back to a head node.

Analysis in this article applies the *gamma lasso* algorithm of Taddy (2014), wherein weights ω_j diminish as a function of $|\hat{\varphi}_j|$.[4] In particular, along the grid of λ_t squelch values,

$$(9)\quad \omega_{jk}^{t} = \left(1 + \gamma |\hat{\varphi}_{jk}^{t-1}|\right)^{-1} \qquad \text{for } \gamma \geq 0.$$

This includes the standard lasso at $\gamma = 0$. For $\gamma > 0$ it provides *diminishing bias* regularization, such that strong signals are less shrunk toward zero than weak signals. This yields sparser $\hat{\boldsymbol{\varphi}}$, which reduces storage and communication needs, and can lead to lower false discovery rates. In practice, a good default is $\gamma = 0$ (i.e., the lasso), but if that provides solutions that are implausibly (or inconveniently) dense, one can experiment with increasing γ.[5]

For selection along the path, we minimize a *corrected AIC* [Hurvich and Tsai (1989)]:

$$(10)\quad \text{AICc:} \quad -2l(\hat{\alpha}_j, \hat{\boldsymbol{\varphi}}_j) + 2df_j \frac{n}{n - df_j - 1},$$

where df_j is the estimated *degrees of freedom* used to fit $\{\hat{\alpha}_j, \hat{\boldsymbol{\varphi}}_j\}$. This corrects the AIC's tendency to over-fit, and Taddy (2014) finds that AICc performs well

[4]The iteratively reweighted least squares algorithm in Section 6 of Taddy (2014) applies directly to Poisson family regressions by setting each iteration's "observation weights" $e^{\eta_{ij}}$ and "weighted response" $\eta_{ij} + c_{ij}/e^{\eta_{ij}} - 1$.

[5]All of our results use the `gamlr` implementation in `R`. The glass-shard example of Section 3 sets $\gamma = 0$ for direct comparison to a lasso penalized alternative, while the Yelp fits of Section 4 all use $\gamma = 1$ for more sparsity.

with the gamma lasso. In Section 3, where computation costs are very low, we also consider CV selection rules: both CV1se, which chooses the largest λ_t with mean OOS deviance no more than one standard error away from minimum, and CVmin, which chooses λ_t at lowest mean OOS deviance.

See Taddy (2014) for much more detail on these techniques. That article reviews diminishing bias regularization by emphasizing its close relationship to weighted L_1 penalization.

3. Example: Glass shards and a parallel softmax. Our motivating big-d applications have the characteristic that m_i is random, and usually pretty big. For example, text mining m_i is the total word count in document i, and web analysis m_i would be the total count of sites visited by a browser. A Poisson model for m_i is not farfetched. However, we also find that DMR also does well in the more common *softmax classification* setting, where $m_i = 1$ always. It thus provides an everyday speedup for classification tasks: even with small-d response categories, you'll be able to fit the model almost d times faster in distribution.[6] Thus, before moving to our Yelp case study, we look at the surprisingly strong performance of DMR in a simple classification problem.

This example considers the small *forensic glass* data set from Venables and Ripley (2002), available in the `MASS` library for `R` under the name `fgl`.[7] The data are 214 observations on shards of glass. The response of interest is of 6 glass types: window float glass (`WinF`), window nonfloat glass (`WinNF`), vehicle window glass (`Veh`), containers (`Con`), tableware (`Tabl`), and vehicle headlamps (`Head`). Covariates for each shard are their refractive index and %-by-weight composition among 8 oxides. Figure 1 shows Poisson regression regularization paths for each glass type, with AICc selection marked by a vertical dashed line.

The response here is a single category, such that $m_i = 1$ and $\hat{\mu}_i = 0$ for all i. This violates the assumption of Poisson generation: $m_i = 1$ is not random. For example, Figure 2 shows the conditional MLE $\mu_i^\star = \log(m_i / \sum_j e^{\hat{\alpha}_j + \mathbf{v}_i' \hat{\boldsymbol{\varphi}}_j})$ at AICc selected coefficients. The result is distributed around, but not equal to, the assumed plug-in of $\hat{\mu}_i = 0$ for all i. However, `dmr` still works: Figure 3 shows the distribution for OOS error in a 20-fold OOS experiment, either using AICc or CV selection *on each individual Poisson regression*, against CV selected models from a lasso path for full multinomial logistic (softmax) regression as implemented in the `glmnet` package for `R` [Friedman, Hastie and Tibshirani (2010)]. There are subtle differences (e.g., AICc DMR selection has lower mean deviance with higher variance), but the full multinomial fits (`glmnet`) do not have any clear advantage over the nearly d-times faster approximation (`distrom`).

[6] In shared-memory parallelization we observe speedups close to linear in d, depending upon machine architecture.

[7] For the code used in this example, type `help(dmr)` in `R` after loading the `distrom` library.

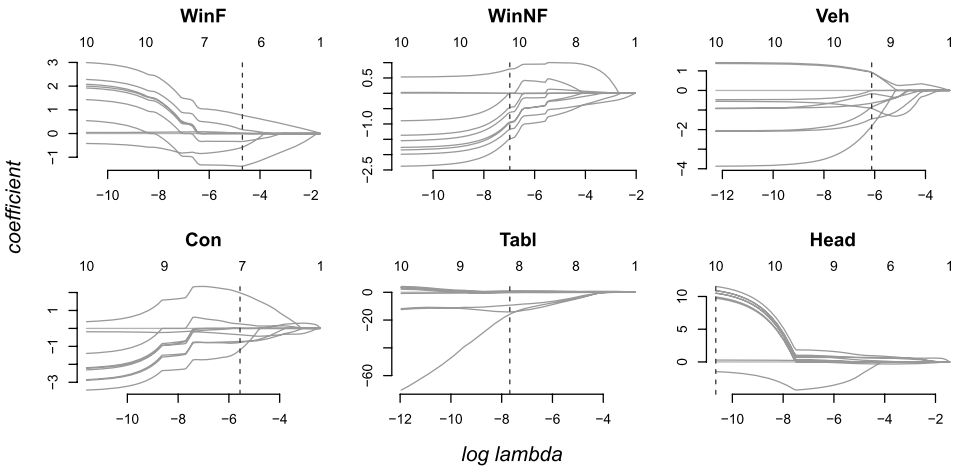

FIG. 1. *Forensic glass. Regularization paths for each glass-type, with AICc selections marked.*

4. Yelp case study. These data were supplied by the review site Yelp for a data mining contest on kaggle.com. The data are available at www.kaggle.com/c/yelp-recruiting/data, and code for processing and estimation is at github.com/TaddyLab/yelp. We consider business, user, and review data sets in the yelp_training_data collection. The reviews, for all sorts of businesses, were recorded on January 19, 2013 for a sample of locations near to Phoenix, AZ. The goal of the competition was to predict the combined number of "funny," "useful," or "cool" (f/u/c) votes that a given review receives from other users. Such information can be used by yelp to promote f/u/c reviews before waiting for the users to grade them as such.

After detailing the data and model in Section 4.1, we describe a series of statistical analyses.

Section 4.2: Investigate model fit under a range of regularization schemes, looking at how word loadings change with the relative weight of penalty on variables of interest vs controls.

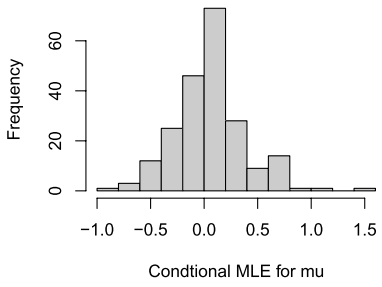

FIG. 2. *Forensic glass. The conditional MLEs $\mu_i^\star$ implied at our DMR coefficient estimates.*

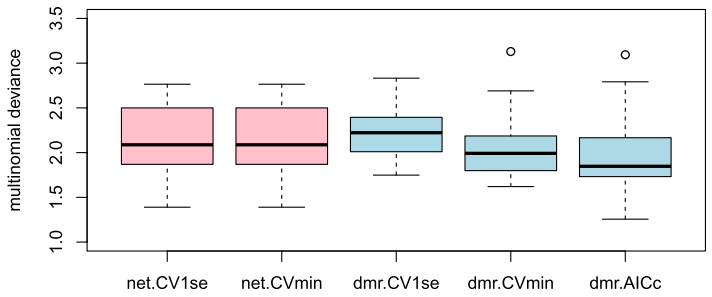

FIG. 3. *Forensic glass. OOS deviance samples in a* 20*-fold OOS experiment. The* net *models are from* `glment` *lasso multinomial logistic regression, and the* dmr *models are our distributed multinomial regression approximation. The applied penalty selection rule is indicated after each model.*

Section 4.3: Use the ideas of "sufficient reduction" to project text through the model onto topics relevant to f/u/c votes or star ratings, and interpret the resulting factor spaces.

Section 4.4: Use the sufficient reductions in prediction of the number of f/u/c votes (i.e., the original `kaggle` task), and compare OOS performance against that of a word-count regression.

Section 4.5: Use the sufficient reductions in treatment effect estimation—for the effect of user experience on rating—while controlling for heterogeneity in review content.

By viewing text data as a big multinomial regression, we are able to address all of the above (and resolve the effects of many collinear attributes on review text) through a single model fit.

4.1. *Data and model specification.* The data are $n = 215{,}879$ reviews on 11,535 businesses by 43,873 users.[8] Review text is split on whitespace and tokenized into words (including combinations of punctuation: potential emoticons). After stripping some common suffixes (e.g., "s," "ing," "ly") and removing a very small set of stopwords (e.g., "the," "and," "or"), we count frequencies for $d = 13{,}938$ words occurring in more than 20 (<0.01%) of the reviews (total word count is $M = 17{,}581{,}214$). Metadata includes review, business, and user attributes:

- `stars`: review star rating (out of 5), from which we subtract the business average rating.
- Review counts for `funny`, `useful`, or `cool` votes. We divide these by the square root of review age, which yields metrics roughly uncorrelated with the posting date.
- `usr.count`: a user's total number of reviews at time of posting the given review.

[8]We have removed reviews with unknown user.

- `usr.stars`: a user's average star rating across all of their reviews.
- A user's average `usr.funny`, `usr.useful`, or `usr.cool` votes per review.
- Business average star rating `biz.stars` and review count `biz.count`.
- Business location among 61 possible cities surrounding (and including) Phoenix.
- Business classification according to Yelp's nonexclusive (and partially user generated) taxonomy. We track membership for 333 categories containing more than 5 businesses.

This yields 405 variables for each review. We also specify random effects[9] for each of the 11,535 businesses, leading to total attribute dimension $p = 11{,}940$. Data components are the $n \times d$ document–term matrix $\mathbf{C}$, the n-vector of row-totals $\mathbf{m}$, and the $n \times p$ attribute matrix $\mathbf{V}$.

We split each row of the attribute matrix into two elements: $\mathbf{a}_i$, the 11 numeric review attributes from `stars` through `biz.count`, and $\mathbf{b}_i$, a length-11,929 vector of dummy indicators for business identity, location, and Yelp classification. This is done to differentiate the variables we deem of primary interest ($\mathbf{a}_i$) from those which we include as controls ($\mathbf{b}_i$); write $\mathbf{V} = [\mathbf{A} \;\; \mathbf{B}]$ as the resulting partition. Columns of $\mathbf{A}$ are normalized to have mean zero and variance one. The multinomial regression of (1) is adapted by similarly splitting each $\boldsymbol{\varphi}_j = [\boldsymbol{\varphi}_j^a, \boldsymbol{\varphi}_j^b]$ and rewriting category intensities $\eta_{ij} = \alpha_j + \mathbf{a}_i'\boldsymbol{\varphi}_j^a + \mathbf{b}_i'\boldsymbol{\varphi}_j^b$.

4.2. *Multinomial model fit and interpretation.* Following the recipe of Section 2.2, each word's Poisson regression is estimated

$$(11)\qquad \hat{\alpha}_j, \hat{\boldsymbol{\varphi}}_j = \underset{\alpha_j, \boldsymbol{\varphi}_j}{\arg\min} \left\{ l(\alpha_j, \boldsymbol{\varphi}_j) + n\lambda \left[\sum_k \omega_{jk}^a |\varphi_{jk}^a| + \frac{1}{\tau} \sum_k \omega_{jk}^b |\varphi_{jk}^b| \right] \right\},$$

where $l(\alpha_j, \boldsymbol{\varphi}_j) = \sum_{i=1}^n [m_i e^{\alpha_j + \mathbf{a}_i'\boldsymbol{\varphi}_j^a + \mathbf{b}_i'\boldsymbol{\varphi}_j^b} - c_{ij}(\alpha_j + \mathbf{a}_i'\boldsymbol{\varphi}_j^a + \mathbf{b}_i'\boldsymbol{\varphi}_j^b)]$. The *relative penalty weight* $\tau > 0$ controls differential regularization between the target variables and the controls. At larger τ values, there is less penalty on $\boldsymbol{\varphi}_j^b$ and the effect of $\mathbf{b}_i$ on c_{ij} has less opportunity to pollute our estimate for $\boldsymbol{\varphi}_j^a$. That is, $\hat{\boldsymbol{\varphi}}_j^a$ becomes more purely a *partial effect*. At the extreme of $\tau = \infty$, any collinearity with $\mathbf{b}_i$ is to be completely removed from the estimated $\hat{\boldsymbol{\varphi}}_j^a$.

As outlined in the Appendix, counts for the 14k words are partitioned into 256 files. Each file is then read by one of 64 workstations, which itself uses 16 cores in parallel to run through the Poisson regressions. Each individual regression is a full gamma lasso path solution over grids of 100 λ_t squelch values, with weights $\omega_{jk}^{at}, \omega_{jk}^{bt}$ updated as in (9) under $\gamma = 1$, and AICc selected coefficients are then written to file. The entire problem (including the sufficient reduction projection of our next section) takes around 1/2 hour.

[9]We call these random, rather than fixed, effects because they are estimated under a penalty which shrinks them toward an overall mean. They will be estimated and we do not marginalize over them.

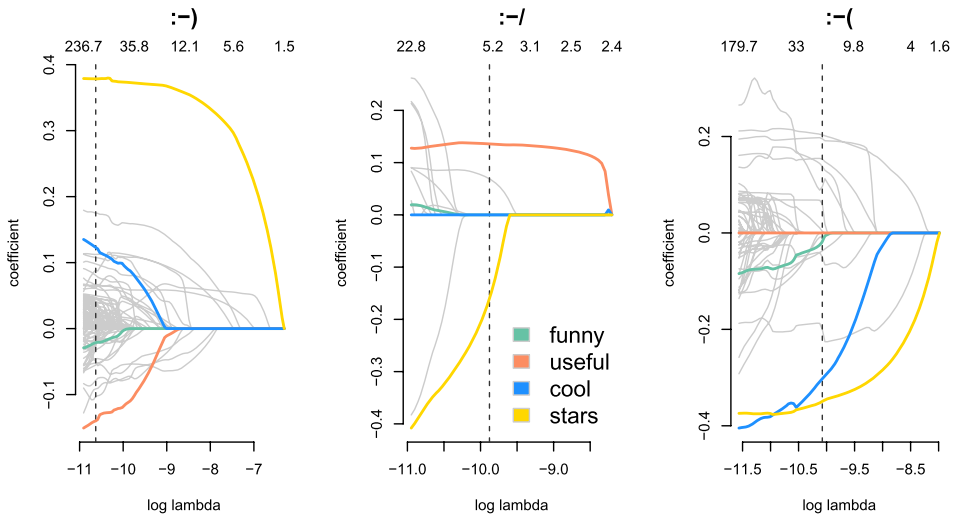

FIG. 4. *Yelp. Poisson regression regularization paths for counts of the tokens* `:-)`, `:-/`, *and* `:-(` *under relative penalty weight* $\tau = 2$. *Coefficient values have been multiplied by the corresponding covariate standard deviation. The legend highlights select covariates, regression degrees of freedom are on the top axis, and our AICc selected estimates are marked with vertical dashed lines.*

Regularization paths for a few of the Poisson regressions, estimated under $\tau = 2$ relative penalty weight, are shown in Figure 4. Coefficient values are scaled to the effect of 1sd change in the corresponding attribute. We see, for example, that at our AICc selection the effect of a 1sd increase in review stars multiplies the expected count (or odds, in the multinomial model) for the happy face `:-)` by around $\exp 0.38 \approx 1.46$, the "hmmm" face `:-/` by $\exp -0.15 \approx 0.86$, and the sad face `:-(` by $\exp -0.35 \approx 0.7$. Notice that `:-/` and `:-(` both occur more often in low-star (negative) reviews, but that `:-/` is associated with useful content, while `:-(` is uncool.

The relative penalty divisor τ allows us to specify the amount of word count variation that is allocated the control variables in $\mathbf{B}$. Such differential penalization is a powerful tool in Big Data analysis, as it allows one to isolate partial effects in messy overdetermined systems. Unfortunately, unlike for λ, we have no objective criterion with which to choose τ. Since it weights the penalty on variables whose effect we would like to *remove* from our targeted coefficients, one could argue that $\tau = \infty$ is the optimal choice. In practice, however, this can lead to inference for the coefficients of interest that is dependent upon only a small subset of documents (since variation in the others is saturated by the controls). We advise using whatever prior knowledge is available to evaluate the appropriateness of results under a range of τ.

For example, Table 1 investigates fit under increasing τ. The numbers of nonzero $\hat{\varphi}^a_{jk}$ (i.e., deemed useful for OOS prediction by AICc) are decreasing with

TABLE 1

Top 10 *words by loading on review characteristics, as a function of relative penalty weight* τ*. The top row for each attribute corresponds to terms ordered by marginal correlations*

	τ	$\hat{\varphi} \neq 0$	**Top ten words by loading**
	Marginal		*Great love amaz favorite deliciou best awesome alway perfect excellent*
+Stars	2	8440	*Unmatch salute :-)) prik laurie pheonix trove banoffee exquisite sublime*
	20	3077	*Heaven perfection gem divine amaz die superb phenomenal fantastic deliciousnes*
	200	508	*Gem heaven awesome wonderful amaz fantastic favorite love notch fabulou*
	Marginal		*Not worst ask horrib minut rude said told would didn*
−Stars	2	8440	*Rude livid disrespect disgrace inexcusab grossest incompet audacity unmelt acknowledge*
	20	3077	*Rude incompet unaccept unprofession inedib worst apolog disrespect insult acknowledge*
	200	508	*Worst horrib awful rude inedib terrib worse tasteles disgust waste*
	Marginal		*You that know like your yelp . . . what don who*
Funny	2	6508	*Dimsum rue reggae acne meathead roid bong crotch peni fart*
	20	1785	*Bitch shit god dude boob idiot fuck hell drunk laugh*
	200	120	*Bitch dear god hell face shit hipst dude man kidd*
	Marginal		*That yelp you thi know biz-photo like all http ://*
Useful	2	5230	*Fiancee rife dimsum maitre jpg poultry harissa bureau redirect breakdown*
	20	884	*biz-photo meow harissa www bookmark :-/ http ://(?), tip*
	200	33	*www http ://com factor already final immediate ask hope*
	Marginal		*Yelp you that biz-photo http ://www know like your*
Cool	2	4031	*Boulder lewi rogue lagunita wanton celebratory hanker politic mozzerella onsite*
	20	577	*Userid htm cen rand poem sultry arlin brimm cubic inspiration*
	200	11	*Biz-photo select yelp along certain fil chose house*

τ for all attributes. This is because **B** accounts for more variation in **C** at higher τ, and there is little residual variation left for **A**. Here, $\tau = 2$ yields top words only indirectly associated with our attributes (e.g., *prik* is positive because Thai food gets high ratings), while full $\tau = \infty$ control leads to near perfect fit and infinite likelihoods conditional on **B** alone. To our eye, $\tau = 20$ manages a good balance: there remain many significant $\hat{\varphi}^a_{jk} \neq 0$, but the model has avoided loading words that are not directly associated with the given attributes. This fit is used in the remainder of our study.

4.3. *Sufficient reduction.* The previous section's coefficient estimates, resolving a complex system of relationships between words and attributes, provide a rich basis for storytelling and exploratory analysis. For many, this is either the end goal or a jumping-off point (e.g., to experiments testing hypotheses generated in explo-

ration). But in our practice, a primary reason for fitting big multinomial models is as a tool for *dimension reduction*, mapping from the original d-dimensional text down to univariate indices that contain all information relevant to a given attribute.

Cook (2007) outlines use of regression models with high-dimensional response as a map to project from that response onto interesting covariates. Taddy (2013a) extends the idea in our context of big multinomials, motivated by applications in text analysis. Both of these articles are focused on *inverse regression* (IR), a technique wherein the fitted model map is applied for prediction of unobserved covariates (e.g., the votes associated with new review text, as in Section 4.4). However, the IR algorithms are prefaced on a more basic concept of *sufficient reduction* (SR), which is useful beyond its application in IR prediction.

Consider observation $\mathbf{c}_i$ from a d-dimensional exponential family linear model, with natural parameter $\boldsymbol{\eta}_i = [\eta_{i1} \cdots \eta_d]'$, $\eta_{ij} = \alpha_j + \mathbf{v}_i \boldsymbol{\varphi}_j$, such that

$$\mathrm{p}(\mathbf{c}_i) = h(\mathbf{c}_i) \exp\bigl[\mathbf{c}_i' \boldsymbol{\eta}_i + A(\boldsymbol{\eta}_i)\bigr], \tag{12}$$

where h is a function of only data (not $\boldsymbol{\eta}_i$) while A is a function of only parameters (not $\mathbf{c}_i$). Both the full multinomial logistic regression model (conditional upon m_i) or our independent Poisson's model (conditional upon $\hat{\mu}_i$) can be written as in (12). Then with $\boldsymbol{\Phi} = [\boldsymbol{\varphi}_1 \cdots \boldsymbol{\varphi}_d]$ the $p \times d$ matrix of regression coefficients, we get

$$\mathrm{p}(\mathbf{c}_i) = h(\mathbf{c}_i) e^{\mathbf{c}_i' \boldsymbol{\alpha}} \exp\bigl[\mathbf{c}_i' \boldsymbol{\Phi}' \mathbf{v}_i + A\bigl(\boldsymbol{\Phi}' \mathbf{v}_i\bigr)\bigr] = \tilde{h}(\mathbf{c}_i) g(\boldsymbol{\Phi} \mathbf{c}_i, \mathbf{v}_i), \tag{13}$$

so that the likelihood factorizes into a function of $\mathbf{c}_i$ only and another function of $\mathbf{v}_i$ that depends upon $\mathbf{c}_i$ only through the projection $\boldsymbol{\Phi} \mathbf{c}_i$. This implies that, conditional upon the regression parameters, $\boldsymbol{\Phi} \mathbf{c}_i$ is a *sufficient statistic* for $\mathbf{v}_i$. That is, $\mathbf{v}_i \perp\!\!\!\perp \mathbf{c}_i | \boldsymbol{\Phi} \mathbf{c}_i$.

We call $\mathbf{z}_i = \boldsymbol{\Phi} \mathbf{c}_i$ an SR *projection*. In practice, we work with estimated SR projections $\mathbf{z}_i = \hat{\boldsymbol{\Phi}} \mathbf{c}_i$ and hope that $\hat{\boldsymbol{\Phi}}$ has been estimated well enough for $\mathbf{z}_i$ to be a useful summary [see Taddy (2013b) for discussion]. In that case, $\hat{\boldsymbol{\Phi}}$ provides a linear map from text into the p-dimensional attribute space. This works just like the *rotation* matrix from common principal components analysis except that, instead of mapping into latent factors, $\boldsymbol{\Phi}$ projects into observed attributes. The resulting $\mathbf{z}_i$ are model-based sufficient statistics, useful in the same roles as a traditional sufficient statistic (like $\bar{x}$). For example, to predict v_{ik} from $\mathbf{c}_i$ we can work with univariate z_{ik} instead of the d-dimensional original text. In general, SR projections are a simple way to organize information in Big Data systems. When new text $\mathbf{c}_i$ arrives, one need just feed it through $\boldsymbol{\Phi}$ to obtain $\mathbf{z}_i$ indices which can be summarized, plotted, and analyzed as desired.

It is important to emphasize that, since estimated loadings $\hat{\varphi}_{ik}$ are partial effects (influence of other attributes has been controlled for), z_{ik} will also correspond to partial rather than marginal association. As another way to see this, note that the factorization in (13) is easily manipulated to show sufficiency for each individual z_{ik} conditional on $\mathbf{v}_{i,-k}$, our vector of attributes omitting the kth. Thus, SR

Correlation matrices

Attributes (**v**)

	f	u	c	⋆
f	1	0.7	0.8	0
u	0.7	1	0.9	0
c	0.8	0.9	1	0
⋆	0	0	0	1

Text projections (**z**)

	f	u	c	⋆
f	1	−0.1	−0.7	−0.4
u	−0.1	1	0.1	−0.2
c	−0.7	0.1	1	0.5
⋆	−0.4	−0.2	0.5	1

FIG. 5. *Correlation for the original review attributes in* **v** (*left*) *and for* **z** (*right*) *SR text projection. Here* f *denotes either the observed* ($v_{\texttt{funny}}$) *or sufficient reduction for* ($z_{\texttt{funny}}$) *the number of funny votes per square root review age, similarly with* u *for useful votes and* c *for cool votes, and* $\star$ *denotes the observed and SR for the number of review stars.*

reduces dimension into a space of information *directly* relevant to an attribute of interest, where influence of text variation due to other attributes has been removed or minimized. Consider the correlation matrices in Figure 5. The original vote attributes are highly positively correlated, while the text projections are either nearly independent (e.g., `useful` against either `funny` or `cool`) or strongly negatively correlated (`funny` and `cool`). This suggests that there are underlying factors that encourage *votes in any category*; only after controlling for these confounding factors do we see the true association between f/u/c content. Similarly, all vote attributes are uncorrelated with star rating, but for the text projections we see both negative (`funny`, `useful`) and positive (`cool`) association.

The three 50–100 word reviews in Figure 6 provide further illustration. A single review (bottom) of a historical site scores highest in `funny` and `useful` attributes (and also in `cool`). The review is neither dry nor useless, but we imagine its high vote count has been influenced by other factors, for example, the page is heavily viewed or people who read reviews of national parks are more likely to vote. In contrast, the two reviews identified through SR projections as having the most funny or useful content appear to us as more directly related to these concepts. The funny review, for a pizza restaurant, is a fictional comedic story. The useful review contains a high proportion of business photos (*biz-photo*), which the multinomial model has identified as directly useful. The machine-learned text projections are able to detect humor and helpfulness distinct from the other factors that lead to increased user votes.

4.4. *Inverse regression for prediction.* Multinomial-based SR projections were originally motivated by Taddy (2013a) for their use in *multinomial inverse regression* [MNIR; see also Taddy (2013c)]. Say v_{iy}, some element of the attribute vector $\mathbf{v}_i$, is viewed as a "response" to be predicted for future realizations. For example, in the original `kaggle` Yelp contest the goal was to predict $v_{i,\texttt{funny}}$, $v_{i,\texttt{useful}}$, or $v_{i,\texttt{cool}}$—the vote attributes. In such applications, an MNIR routine would use the SR projection into v_{iy}, $z_{iy} = \sum_j \hat{\varphi}_{jy} c_{ij}$, to build a *forward regres-*

Funniest 50–100 word review, by SR projection $\mathbf{z}_{\texttt{funny}}$.

Dear La Piazza al Forno: We need to talk. I do not quite know how to say this so I'm just going to come out with it. I've been seeing someone else. How long? About a year now. Am I in love? Yes. Was it you? It was. The day you decided to remove hoagies from your lunch menu, about a year ago, I'm sorry, but it really was you… and not me. Hey… wait… put down that pizza peel… try to stay calm… please? [Olive oil container whizzing past head] Please! Stop throwing shit at me… everyone breaks up on social media these days… or have not you heard? Wow, what a Bitch!

Most useful 50–100 word review, by SR projection $\mathbf{z}_{\texttt{useful}}$.

We found Sprouts shortly after moving to town. There's a nice selection of Groceries & Vitamins. It's like a cheaper, smaller version of Whole Foods. [biz-photo] [biz-photo] We shop here at least once a week. I like their selection of Peppers…. I like my spicy food! [biz-photo][biz-photo][biz-photo] Their freshly made Pizza is not too bad either. [biz-photo] Overall, it's a nice shopping experience for all of us. Return Factor—100%.

Funniest and most useful 50–100 word review, as voted by Yelp users (votes normalized by square root of review age).

I use to come down to Coolidge quite a bit and one of the cool things I use to do was come over here and visit the ruins. A great piece of Arizona history! Do you remember the Five C's? Well, this is cotton country. The Park Rangers will tell you they do not really know how old the ruins are, but most guess at around 600 years plus. But thanks to a forward thinking US Government, the ruins are now protected by a 70 foot high shelter. Trust me, it comes in handy in July and August, the two months I seem to visit here most. LOL. I would also recommend a visit to the bookstore. It stocks a variety of First Nation history, as well as info on the area. http://www.nps.gov/cagr/index.htm. While you are in Coolidge, I would recommend the Gallopin' Goose for drinks or bar food, and Tag's for dinner. Both are great!

FIG. 6. *Illustration of the information contained in sufficient projections* $\mathbf{z}$. *The top two reviews are those, among all where* $m \in (50, 100)$, *with highest SR projection scores into the* `funny` *and* `useful` *attribute spaces. For comparison, we also show the single* 50–100 *word review with highest values for both* $v_{\texttt{funny}}$ *and* $v_{\texttt{useful}}$ (*recall that these are vote totals per square root review age*). *Note that, since variance of* $\mathbf{z}$ *increases with* m, *high scoring reviews tend to be longer. One can also, as in Taddy* (2013a), *divide the SR projections by document length and work with normalized* z/m. *On this scale, the funniest review is "Holy Mother of God" and the most useful review is "Ask for Nick!"*

TABLE 2

Yelp. Out-of-sample R^2 in prediction for vote attributes (normalized by root review age) in 5-fold CV. The top row shows a standard lasso regression from the vote attribute onto text and all nonvote attributes, while the bottom row holds results for MNIR followed by lasso regression from the vote attribute onto review length (m_i), nonvote attributes, and the corresponding univariate SR projection

	Forward regression		**Average out-of-sample R^2**		
	Input variables	**Dimension**	**`funny`**	**`useful`**	**`cool`**
Standard lasso	Nonvote attributes, $\mathbf{C}$	25,876	0.308	0.291	0.339
MNIR + lasso	Nonvote attributes, $\mathbf{z}$, $\mathbf{m}$	11,940	0.316	0.296	0.341

sion that predicts v_{iy} from z_{iy}, $\mathbf{v}_{i,-y}$ (attributes omitting y), and m_i.[10] This $p+1$ dimensional regression replaces the $d+p-1$ dimensional one that would have been necessary to predict v_{iy} from $\mathbf{v}_{i,-y}$ and $\mathbf{c}_i$, the original text counts.

Estimating an *inverse* regression in order to get at another *forward* regression may seem a strange use of resources. But there are a variety of reasons to consider MNIR. Computationally, through either the techniques of this article or the collapsing of Taddy (2013a), the multinomial regression estimation can occur in distribution on many independent machines. This is useful when the full count matrix $\mathbf{C}$ is too big to fit in memory. Another reason to use MNIR is for statistical efficiency when d is big relative to n. Assuming a multinomial distribution for $\mathbf{c}_i|\mathbf{v}_i$ introduces information into the estimation problem (a less generous term is "bias"). In particular, it implies that each of the $M=\sum_i m_i$ counts are independent observations, such that the sample size for learning $\mathbf{\Phi}$ becomes M rather than n. That is, estimation variance decreases with the number of words rather than the number of documents [see Taddy (2013b)].

As an illustration, Table 2 shows results for prediction of individual f/u/c vote attributes, both through MNIR with lasso forward regression and for a standard lasso onto the full text counts. That is, MNIR fits $\mathbb{E}[v_{iy}]=\beta_0+[\mathbf{v}_{i,-\mathrm{f/u/c}},m_i,z_{iy}]'\boldsymbol{\beta}$ while the comparator fits $\mathbb{E}[v_{iy}]=\beta_0+[\mathbf{v}_{i,-\mathrm{f/u/c}},\mathbf{c}_i]'\boldsymbol{\beta}$, where $\mathbf{v}_{i,-\mathrm{f/u/c}}$ denotes all nonvote attributes. For MNIR each $\hat{\mathbf{\Phi}}$ (hence, z_{iy}) is also estimated using only the training sample, and in both cases prediction rules were selected via AICc minimization along the L_1 regularization path. We see that MNIR forward regression, replacing 13,938 covariates from $\mathbf{c}_i$ with just the two numbers z_{yi} and m_i, does not suffer against the full lasso comparator (indeed, it is very slightly better in each case). Such performance is typical of what we have observed in application.[11] This

[10]The SR result that applies here is $v_{iy}\perp\!\!\!\perp\mathbf{c}_i|z_{iy},\mathbf{v}_{i,-y},m_i$. Since sufficiency for z_{iy} from the multinomial factorization is *conditional upon* m_i, these document totals need to be conditioned upon in forward regression.

[11]In Taddy (2013a), the MNIR routines more significantly outperform lasso comparators in OOS prediction. However, the data sets used in that paper are both very small, with $M\gg n$. Thus, our

is not evidence that the text counts do not matter: each full lasso estimates at least 4000 terms having nonzero coefficients. Rather, the multinomial model is a good enough fit that the factorization results of (13) apply and all relevant information is contained in the SR projection.[12]

Note that the MNIR forward regression ignores projection from text onto any other nonvote attributes. This is because those attributes are conditioned upon in forward regression. Indeed, Taddy (2013a, 2013b) argue that, in prediction for a single variable, you only need fit the multinomial dependence between counts and that single variable. This yields SR projection based on marginal association, which can work as well as that based on partial association for simple predictions. The benefit of fitting models for high-dimensional $\mathbf{v}_i$ is that we are then able to interpret the resulting partial effects and SR projections, as in Sections 4.2–4.3. It is also useful in more structured prediction settings, as in the next section.

4.5. *Confounder adjustment in treatment effect estimation.* In our final application, we illustrate use of SR projections as convenient low-dimensional *controls* in treatment effect estimation. The task here has a particular attribute, say t, whose effect on another, say y, you want to estimate. You want to know what will happen to y if t changes independently from the other attributes. Unfortunately, everything is collinear in the data and both y and t could be correlated to other unobserved confounders. Your best option is to estimate the treatment effect—that of t on y—while controlling for observable potential confounders. In text analysis, this includes controlling for the text content itself.

Consider estimating the effect of a user's experience—the number of reviews that they have written—on their expected rating. That is, are experienced users more critical, perhaps because they have become more discerning? Or do they tend to give more positive reviews, perhaps because community norms encourage a high average rating? It is hard to imagine getting firm evidence in either direction without running a randomized trial—we will always be worried about the effect of an omitted confounder. However, we can try our best and condition on available information. In particular, we can condition on content to ask the question: even given the same review message, would an experienced user give more or less stars than a newbie?

The response attribute, v_{iy}, is *star rating*. The treatment, v_{it}, is the log *number of reviews* by the author (including the current review, so never less than one). Results for estimation of the effect of v_{it} on v_{iy}, conditioning on different control

statistical efficiency argument—that for MNIR estimation variance decreases with M instead of n—is working heavily in favor of MNIR. Here, even though $M > n$, vocabulary size d is smaller than n and linear regression is already plenty efficient.

[12]We have also found success applying nonlinear learning (e.g., trees) in forward regression after SR projection. Methods that are too expensive or unstable on the full text work nicely on the reduced dimension subspace.

TABLE 3

Estimated effect "γ" of user experience (log number of reviews) on number of stars rated. Each corresponds to different levels of confounder adjustment. The effects are all AICc selected estimates along a $\gamma = 10$ (very near to L_0) gamma lasso regularization path, where all of the other regression coefficients were unpenalized. Thus, they are significant, in the sense that the AICc deems v_{it} useful for predicting v_{iy} even after all variation explained by confounders has been removed

	Marginal	**Conditional on attributes only**	**Adding and interacting text SR**
Effect estimate	0.003	0.015	0.020

variables, are detailed in Table 3. A naïve estimate for the effect of experience on rating, estimated through the marginal regression $\mathbb{E}[v_{iy}] = \beta_0 + v_{it}\gamma$, is a $\hat{\gamma} = 0.003$ increase in number of stars per extra unit log review count. Use $\mathbf{v}_{i,-yt}$ to denote all other attributes. Then an improved estimate of the treatment effect is obtained by fitting $\mathbb{E}[v_{iy}] = \beta_0 + v_{it}\gamma + \mathbf{v}'_{i,-yt}\boldsymbol{\beta}$, which yields the much larger $\hat{\gamma} = 0.015$.

Finally, we would like to control for $\mathbf{c}_i$, the review content summarized as word counts. It would also be nice to control for content interacting with attributes since, for example, positive content for a restaurant might imply a different star rating boost than it does for a bowling alley. Unfortunately, interacting 13,938 dimensional $\mathbf{c}_i$ with the 333 business categories yields almost 4.7 million regression coefficients. This is more controls than we have observations. However, the SR projections offer a low-dimensional alternative. Write z_{iy} and z_{it} for the SR projections onto response and treatment, respectively. Then sufficiency factorization implies

$$v_{iy}, v_{it} \perp\!\!\!\perp \mathbf{c}_i | z_{iy}, z_{it}, m_i, \mathbf{v}_{i,-yt}. \tag{14}$$

That is, the joint distribution of treatment and control is independent of the text given SR projection into each. This suggests we can control for review content, and its interaction with business classification, simply by adding to our conditioning set $[z_{iy}, z_{it}, m_i]$ and its interaction with business classification. The resulting regression, with around 13k control coefficients instead of 4.7 million, yields the still larger treatment effect estimate $\hat{\gamma} = 0.02$.

5. Discussion. Distributed estimation for multinomial regression allows such models to be applied on a new scale, one that is limited only by the number of machines you are able to procure. This is an important advance not only for our motivating text analysis applications, but also for any other setting of high-dimensional multinomial modeling. This includes any softmax classification model.

One message of this paper has been that Poisson factorization enables fast estimation of multinomial distributions. It has been pointed out to us that, in unstructured data analysis, a Poisson seems little more arbitrary than a multinomial

model. Equation (2) clarifies this issue: the only additional assumption one makes by working with independent Poissons is that the aggregate total, m_i, is Poisson. We have attempted to mitigate the influence of this assumption, but that is unnecessary if you consider the Poisson a fine model in and of itself.

Finally, we wish to emphasize the relative simplicity of this approach. Although this article describes models for complex language systems, and it may not seem to the reader that we are providing anything "simple," almost all of this material is *just Poisson regression.* We have used the ideas of Taddy (2014) and the gamma lasso to fit these regressions, but any generalized linear model estimator could have been applied. As implemented in this article, there are only two tuning parameters in the entire system: the gamma lasso weight γ, which can be safely fixed at zero (for lasso regression) as a solid default; and the relative confounder-penalty divisor τ. Specification of τ is a clearly subjective choice,[13] but such subjectivity is inevitable in any structural inference that does not involve a random or pseudo-random experiment.

Too often, social scientists faced with text data will jump to latent space models (e.g., topic models) as the first step in their analysis. Unless the phenomena that they'd like to measure is a dominant source of variation in word choice, these latent topics will be mostly irrelevant. The same scientist faced with more familiar response variables—such as money spent—would likely have used regression modeling with a mix of observable covariates and fixed or random effects, instead of trying to model any sort of latent space. Thus, without wanting to claim that topic and related models are not useful (they are very useful), we hope that this article will give social scientists the option of using the same type of regression tools for text analysis that they use successfully in their nontext research.

APPENDIX: MAPREDUCE DETAILS

MapReduce (MR) is a recipe for analysis of massive data sets, designed to work when the data itself is distributed: stored in many files on a network of distinct machines. The most common platform for MR is Hadoop paired with a distributed file-system (DFS) optimized for such operations (e.g., Hadoop DFS).

A MapReduce routine has three main steps: map, partition, and reduce. The partition is handled by Hadoop, such that we need worry only about map and reduce. The map operation parses unstructured data into a special format. For us, in a text mining example, the mapper program will take a document as input, parse the text into tokens (e.g., words), and output lines of processed token counts: "`token document|count`." The pre-tab item (our `token`) is called a "key." Hadoop's sort facility uses these keys to send the output of your mappers to machines for the next step, reducers, ensuring that all instances of the same key (e.g., the same word) are grouped together at a single reducer. The reducer then executes some

[13]Except for when you have enough data to identify the model with $\tau = \infty$.

Algorithm 1 MapReduce DMR

Map: For each document, tokenize and count sums for each token. Save the total counts m_i along with attribute information $\mathbf{v}_i$. Output `token document|count`.

Combine totals m_i and attributes $\mathbf{v}_i$ into a single table, say `VM`. This info can be generated during map or extracted in earlier steps. Cache `VM` so it is available to your reducers.

Reduce: For each token key "j," obtain a regularization path for Poisson regression of counts c_{ij} on attributes $\mathbf{v}_i$ with $\hat{\mu}_i = \log m_i$. Apply AICc to select a segment of coefficients from this path, say $\hat{\boldsymbol{\varphi}}_j$, and output nonzero elements in sparse triplet format: `word|attribute|phi`.

Each reducer writes coefficients $\hat{\boldsymbol{\varphi}}_j$ of interest to file, and maintains a running total for SR projection, $\mathbf{z}_i \mathrel{+}= \mathbf{c}_i' \hat{\boldsymbol{\varphi}}_j$, output as say `Z.r` for the rth reducer. When all machines are done we aggregate `Z.r` to get the complete projections.

operation that is independent-by-key, and the output is written to file (usually one file per reducer).

DMR fits nicely in the MR framework. Our map step tokenizes your unstructured data and organizes the output by token keys. Reduce then takes all observations on a single token and runs a Poisson log regression, applying the gamma lasso with IC selection to obtain coefficient estimates. This recipe is detailed in Algorithm 1.

We have written this as a single MR algorithm, but other variations may work better for your computing architecture. Our most common implementation uses Hadoop to execute the map on a large number of document files, but replaces the regression reduce step with a simple write, to solid state storage "midway" at the University of Chicago's Research Computing Center, of token counts tabulated by observation. For example, given 64 reducer machines on AWS, the result is 64 text tables on midway, with lines "`word|doc|count`," each containing *all* nonzero counts for a subset of the vocabulary of tokens. These files are small enough to fit in working memory[14] and can be analyzed on distinct compute nodes, each employing another layer of parallelization in looping through Poisson regression for each token. This scheme is able to take advantage of Hadoop for fast tokenization of distributed data, and of high performance computing architecture (much faster than, say, a virtual AWS instance) for each regression. It is a model that should work well for the many statisticians who have access to computing grids designed for high throughput tasks more traditionally associated with physics or chemistry.

[14]If not, use more reducers or split the files.

REFERENCES

BIRCH, M. W. (1963). Maximum likelihood in three-way contingency tables. *J. Roy. Statist. Soc. Ser. B* **25** 220–233. MR0168065

COOK, R. D. (2007). Fisher lecture: Dimension reduction in regression. *Statist. Sci.* **22** 1–26. MR2408655

DEAN, J. and GHEMAWAT, S. (2008). MapReduce: Simplified data processing on large clusters. *Comm. ACM* **51** 107–113.

FRIEDMAN, J., HASTIE, T. and TIBSHIRANI, R. (2010). Regularization paths for generalized linear models via coordinate descent. *J. Stat. Softw.* **33** 1–22.

GOPALAN, P., HOFMAN, J. M. and BLEI, D. M. (2013). Scalable recommendation with Poisson factorization. Available at arXiv:1311.1704.

HODGES, J. L. JR. and LE CAM, L. (1960). The Poisson approximation to the Poisson binomial distribution. *Ann. Math. Statist.* **31** 737–740. MR0117812

HURVICH, C. M. and TSAI, C.-L. (1989). Regression and time series model selection in small samples. *Biometrika* **76** 297–307. MR1016020

MCDONALD, D. R. (1980). On the Poisson approximation to the multinomial distribution. *Canad. J. Statist.* **8** 115–118. MR0595721

PALMGREN, J. (1981). The Fisher information matrix for log linear models arguing conditionally on observed explanatory variables. *Biometrika* **68** 563–566. MR0626420

TADDY, M. (2013a). Multinomial inverse regression for text analysis. *J. Amer. Statist. Assoc.* **108** 755–770. MR3174658

TADDY, M. (2013b). Rejoinder: Efficiency and structure in MNIR. *J. Amer. Statist. Assoc.* **108** 772–774. MR3174661

TADDY, M. (2013c). Measuring political sentiment on Twitter: Factor optimal design for multinomial inverse regression. *Technometrics* **55** 415–425. MR3176547

TADDY, M. (2014). One-step estimator paths for concave regularization. Available at arXiv:1308.5623.

TIBSHIRANI, R. (1996). Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B* **58** 267–288. MR1379242

VENABLES, W. N. and RIPLEY, B. D. (2002). *Modern Applied Statistics with S*, 4th ed. Springer, New York.

UNIVERSITY OF CHICAGO
BOOTH SCHOOL OF BUSINESS
5807 SOUTH WOODLAWN AVENUE
CHICAGO, ILLINOIS 60637
USA
E-MAIL: taddy@chicagobooth.edu