STAT W4249 - Final Project

Yiran Dong, Chang Liu, Rongyao Huang, Gregory Werbin

05/11/2014

"Done is better than perfect."

—poster in the cafeteria at Facebook's NY office.

1 Introduction

Among the many things that haunt a graduate student's mind, finding a job is probably the most important and monstrous. While an ideal job is the intersection of three sets - what one loves, what one is good at, and what the society values - an answer to the third question is enough to guarantee a good pay.

For this research project, our major task is to predict job salary based on texts that describe the job. The data set we use includes NYC government job postings from June 2012 up till now. It has information on salary, the agency/department that hires, job title, job description, job level, job requirements, etc.

The general approach is to perform topic modeling on the collection of texts, reveal the latent topics as explanatory variables, and then predict the salary with different models. In section 2, we lay down the theoretical foundation for both topic models and predictive models. We specifically highlight the difference between the Latent Dirichlet Allocation (LDA) and the Structural Topic Model (STM). In section 3, we describe how we clean and process our data, and how we construct and select the variables. In section 4, we present our exploratory analysis, which give insights to modeling choices. In section 5, we discuss results from our topic models and predictive models, compare their performances based on correlation, R-square, distance correlation, accuracy, sensitivity and specificity, and provide possible interpretations. Finally, in section 6, we conclude that our model exhibits modest performance on salary as a continuous variable, but does well when the target is widened to a binary variable.

2 Theoretical Approaches

2.1 Topic Model

The difficulty with text-based data is that a lot of information of interest is buried in the text itself and cannot be analyzed directly. In this paper, we attempt to use the text of a job posting to predict the salary associated with that posting. Our general approach is to assume that each posting contains several latent "topics," and that some topics are associated with higher salaries, other topics are associated with lower salaries, and that there is not substantial overlap between the two categories. We develop a "topic score" that measures how strongly a particular topic is represented in each document, and use these scores to predict salary.

We start with a topic model that imputes a topic for each individual word. Topic models typically assume that the entire collection of documents—called the "corpus"—is a sequence of random variables, and that the realization of each random variable is a particular word. The set of all distinct words in the corpus is called the "vocabulary."[1] Each word is labeled to indicate the document it comes from, and words within

[1] For clarity, we will always refer to distinct words from the vocabulary as "vocabulary words" or "vocabulary elements." The term "words" will refer to the random variables themselves. Therefore the "vocabulary" is the support the random variables, and a "vocabulary word" is a possible realization.

documents are assumed to be exchangeable. Therefore we model the entire corpus as a sequence of clustered or grouped random variables.

The core feature of a topic model is the assumption that each topic corresponds to a different categorical distribution over the vocabulary. Therefore each word must be assigned a topic, so each word's distribution over the vocabulary is known conditional on that word's topic. The topics, however, are latent, and so cannot be assigned to words in a deterministic way. Therefore we further assume that the topic of a word is also random, and that the distribution over topics is the same for every word in a document.

The likelihood for each word is thus a mixture of K distributions over the vocabulary, $\{\phi_k\}_{k=1}^K$, with mixture weights $\theta_{i(n)}$. A topic can then be "defined" as a distribution over the vocabulary. Since θ_i is the expected frequency of topics in document i, we select each θ_i as the vector of topic scores we will use to predict salary.

We consider two models for θ and ϕ. The first, known as "latent Dirichlet allocation" (LDA), assumes that $\theta \sim Dirichlet(\alpha)$ and $\phi \sim Dirichlet(\beta)$, where α and β are fitted with the model. This is implemented in R with the `LDA()` function in package `topicmodels`. The second is a "structural topic model" (STM), in which the prior distributions for θ and ϕ depend on document-level covariates. The STM is implemented in R package `stm` as function `stm()`.

The STM assumes $\theta_i \sim LogitNormal(\mu_i, \Sigma_\theta)$, where $(\mu_i)_k \equiv X_i'\gamma \ \forall k$, X_i is a vector of document-level covariates, and γ is a vector of coefficients. Unlike in LDA, ϕ in STM is a deterministic function of a categorical covariate Z_i that takes values $g \in \{1, \dots G\}$, so that $g(i)$ is the level of Z observed for document i. Then $\phi_i \propto exp(\kappa + \kappa_g(i) + \kappa_k + \kappa_{g(i),k}$, where κ is the overall frequency of vocabulary words in the corpus and $\kappa_g(i)$ is the deviation from that frequency in documents with $Z_i = g(i)$. κ_k is the theoretical deviation from that frequency among words with topic $t_n = k$, and $\kappa_{g(i),k}$ is an interaction term. $\kappa_{g(i),k}$ and Σ_theta are given regularizing, sparsity-inducing priors.

The following is a side-by-side comparison of the data-generating processes for LDA and STM. The two models are drawn as plate diagrams in Figures 1 and 2.

Latent Dirichlet Allocation

1. For each document i, draw $\theta_i \sim Dirichlet(\alpha)$
2. For each topic k, draw $\phi_k \sim Dirichlet(\beta)$
3. For each word n, draw $t_n \sim categorical(\theta_{i(n)})$
4. For each word n, draw $w_n \sim categorical(\phi_{t_n})$

Structural Topic Model

1. For each i, draw $\theta_i \sim LogitNormal(\mu_i, \Sigma_\theta)$
2. For each topic k, $\phi_k \propto exp(\kappa + \kappa_g(i) + \kappa_k + \kappa_{g(i),k}$
3. For each word n, draw $t_n \sim categorical(\theta_{i(n)})$
4. For each word n, draw $w_n \sim categorical(\phi_{t_n})$

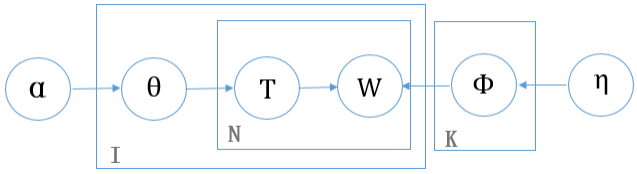

Figure 1: Plate Diagram for the Latent Dirichlet Allocation

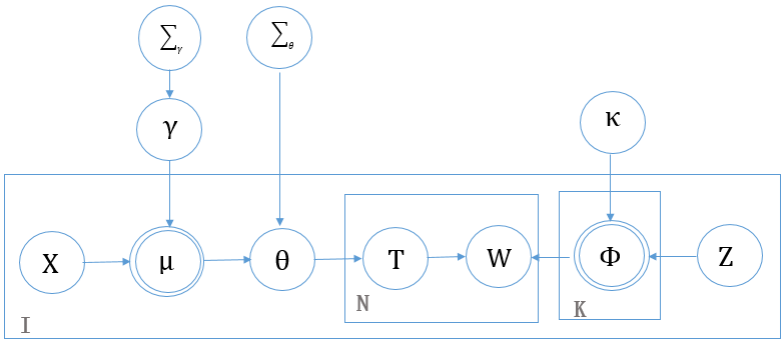

Figure 2: Plate Diagram for the Structural Topic Model

2.2 Predictive Model

LDA and STM are generative models for words, not for documents, but they fit parameters on the document level. In this case, the parameters are probability distributions over latent topics, but we re-interpret each probability as the extent to which a document is associated with a particular latent topic. This interpretation is crude but intuitive, and directly provides us with a relatively low-dimensional set of features. Fitting a prediction model on these features is analogous to fitting a prediction model on principal components or some other reduced feature space.

We first consider an OLS linear regression of salary midpoint on θ. We then split salary at the median into a binary response, to help reduce both noise and clumping in the continuous data. We then fit a linear classifier, a logistic regression, and a support vector machine. For each model, the predictors consist of θ and the variables described in Section 3. We also fit each model on θ alone to determine the extent to which the document metadata, rather than the document text, is responsible for any predictive success.

3 Data

We fit this model to the NYC Jobs data set from the NYC Open Data Portal[2]. It contains the text content and complete metadata for 1,658 job postings from the official City of New York jobs website. It is continuously updated; we use postings last updated on or before April 22, 2014. The dataset contains 26 variables, and only 865 observations are unique; the rest are duplicates, because many jobs were posted separately to internal and external sites.

3.1 Variable Construction

In the original data set we have four text variables: "Job Description," "Minimum Qual Requirements," "Preferred Skills," and "Additional Information." For each posting, these were concatenated to form a single document. We have only the lower and upper bound of the salary offering. For our analysis we use the average of the lower and upper bounds—the midpoint of offered salaries—as the response variable. Salary also has three frequency bases: hourly, daily or annually. Separate variables were indicates the number of hours per shift and hours per week, which we used to standardize all salaries to the annual basis. However several were missing, and for these we assumed a 40-hour workweek and 50 working weeks in a year. This assumption might be invalid for daily-pay jobs. Table 1 shows that daily-pay jobs have high annualized salaries, but bear titles like "machinist's helper" and "oiler." These could be well-paying union positions, but are unlikely to pay $90,801 and $130,980 a year, respectively.

We discretize salary by splitting it at the median, and identifying salaries above and below the median with 1 and 0, respectively. The agency associated with each job posting was binned from 39 categories into 6, which were given heuristic names: "finance," "infrastructure," "social services," "law," "security," and "information technology." A text variable called "Residency Requirement" was recoded to indicate whether

[2] Available at `https://data.cityofnewyork.us/Business/NYC-Jobs/kpav-sd4t`

1 2

Table 1: Aasf.

New York City residency was required. The length in words was calculated for each concatenated posting, as well as the Flesch-Kincaid Grade Level score.[3]

Function `textProcessor()` from package `stm` was used to remove punctuation and stop words, and to stem each word. Some technical difficulties arose during this stage and a few garbled words were produced. These words were not omitted before fitting the topic models. We argue that this is most likely to have decreased predictive performance, and therefore that our results remain valid.

3.2 Variable Selection

The X variables in the STM model, as defined in Section 2.1, are called "prevalence" variables because they determine the prevalence of each topic in each document. We use both agency and level as prevalence variables. For parsimony and to preserve degrees of freedom, only agency was used for the Z variable. This is known as the "content" variable, along which word frequency deviates conditional on topic. These were chosen intuitively, and and by eliminating other variables that would be implausible in these roles. We use level, residency requirement, and length of text as predictors, as well as the fitted θ values.[4]

4 Exploratory Analysis

We first seek to understand the distribution of our response variable, the salary offered with each job posting, and how it relates to the other variables under consideration. The median salary is \$69,930 and the distribution of jobs paying \$180,000 and below does not exhibit notable skew. However, there is one job posting that offers a midpoint of \$404,600, for a "Chief Consulting Psychologist." It is obvious from the text that this posting is for a very experienced individual with a medical degree, and therefore that the off-the-charts salary is not a mistake or an outlier. We want to avoid throwing out data, but at the same time want to ensure that no one data point skews the results excessively. Transforming salary to a log scale keeps this point within a reasonable range of the others, at the cost of imposing a negative skew on the data as shown in Figure 3.

Figure 3 also suggests that "M-level" jobs generally fall at the high end of the salary distribution. The civil service titles of such jobs all indicate managerial positions—the word "manager" appears in 86 of them, and the word stem "admin" appears in 113, out of 198 total. The agency associated with the job does not exhibit a similar pattern, although there is clear variation between agencies.

[3] Reading score is defined as $0.39(\frac{totalwords}{totalsentences}) + 11.8(\frac{totalsyllables}{totalwords}) - 15.59$ and was computed with R package `koRpus`.

[4] Blei and (20??) suggest fitting the model to estimated empirical topic frequencies rather than their theoretical means, i.e. θ. This is much more computationally expensive and would need to be implemented from scratch.

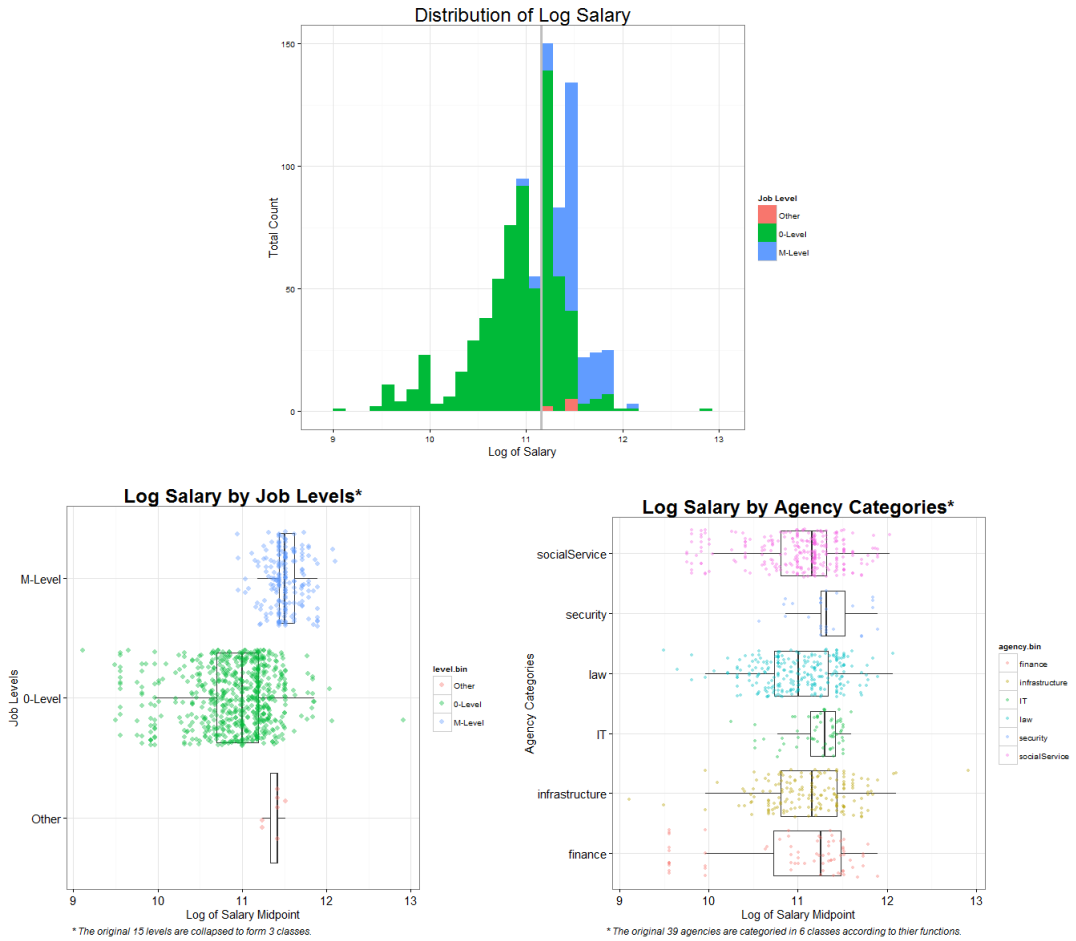

Figure 3: Distribution of log salary, by level. Gray line is the median.

We are ultimately interested in modeling salary as a function of the document text. Therefore we would like to find a way to explore the relationship graphically before attempting to build a model. Text data, even in our simplified multinomial model, has far too many dimensions to visualize directly. We use classical multidimensional scaling to "compress" the document space from $\mathbb{Z}_+^{|V|}$ to $\mathbb{R}^2$ and the result is visualized in Figure 4.

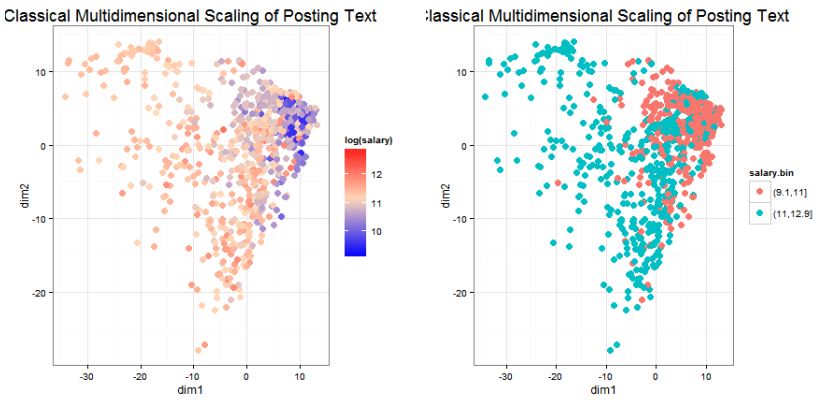

Figure 4: Classical MDS, colored by salary.

In Section 3 we intuited that agency and level would make the most sense as covariates in the STM. Figure 5 is the same MDS plot, colored by agency and level. Neither factor is separating in the reduced feature space. However, there does appear to be an underlying structure. For instance, M-level jobs only appear in the bottom part of the "cone," and there is a thick band of social service jobs in a similar region.

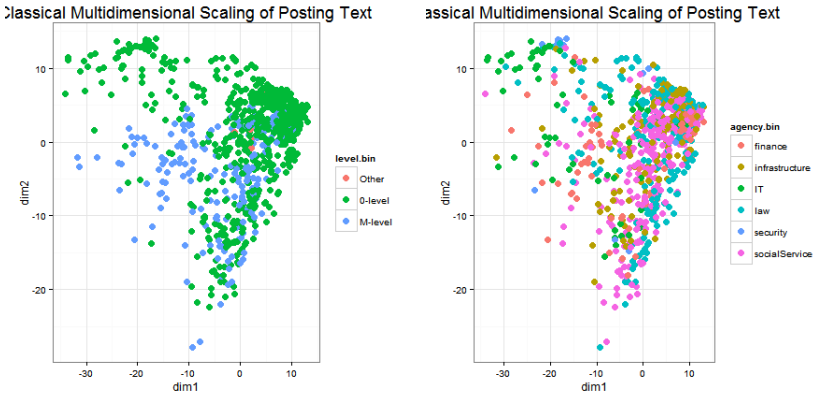

Figure 5: Classical MDS, colored by level and agency.

Finally, we would like to get a sense of whether level and agency are associated with the other predictors we plan to use for salary, since level and agency will be included as predictors as well. For example, Figure 6 shows that job level is completely unrelated to both text length and reading score. Most typical prediction models perform better with uncorrelated inputs, so this is a desirable result. Moreover, text length exhibits a small positive association with salary, so we feel justified in including it among the prediction covariates.

Figure 6: Classical MDS, colored by level and agency.

In addition to the previous analysis , we want to get a sense of how average salary distributed among different locations. Figure 7 shows that agencies are mostly located in Manhattan and Queens, and agencies with high-salary job postings in Manhattan. There are fewer agencies in Bronx and Staten island, but a very large portion of the job postings by the agencies located in the latter ensure payments falling in the $78875 - 81147$ and $81157 - 92845$ categories. Since we found no explicit correlation between salary and locations from this map, we decided to exclude locations from our covariates in our final regression model.

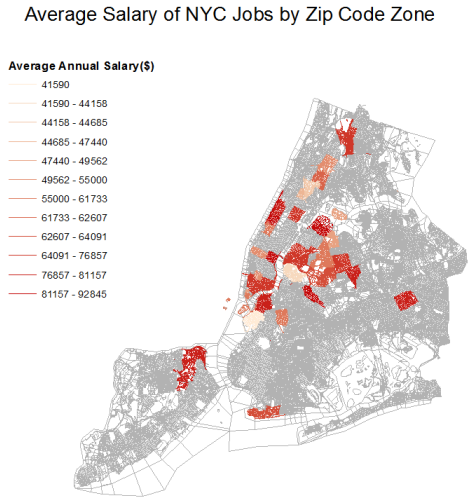

Figure 7: Average Salary of NYC jobs by Zip Code Zone.

5 Results

5.1 Choosing K

We decide the optimal number of topics using the measure proposed by Arun, Suresh, Madhavan, & Murty (2010). They regard topic models as matrix factorization mechanisms, wherein a given corpus C is split into

two matrix factors M_1 $(T \times W)$ and M_2 $(D \times T)$, where D is the number of documents contained in the corpus, W is the size of the fixed vocabulary, and T is the supposedly right number of topics. They propose a measure that computes the symmetric Kullback-Leibler divergence of the singular value distributions of M_1 and the distribution of the vector $L \cdot M_2$, where L is a D-vector containing the lengths of each document in the corpus. They show that under certain conditions the proposed measure reliably reaches a local minimum around an optimal number of topics. The measure is calculated as follows:

$$\text{objective}(M_1, M_2) = \text{KLdiv}(C_{M_1}, C_{M_2}) + \text{KLdiv}(C_{M_2}, C_{M_1}) \tag{1}$$

where C_{M_1} is the distribution of singular values of M_1 and C_{M_2} is the distribution obtained by normalizing the vector $L \cdot M_2$.

The optimal number of topics under this criterion is the one in which topics are as separated or exclusive as possible, and this is a desirable outcome for building a prediction model. Least squares regression is most effective on relatively uncorrelated predictors, and linear classifiers perform better on separated data. In principle, topics with more exclusive words are more informative. Highly separated topics could also have clearer interpretations. K was optimized separately for LDA and STM, and $K = 20$ was chosen as optimal for both. The traces of the objective functions are shown in Figure 8.

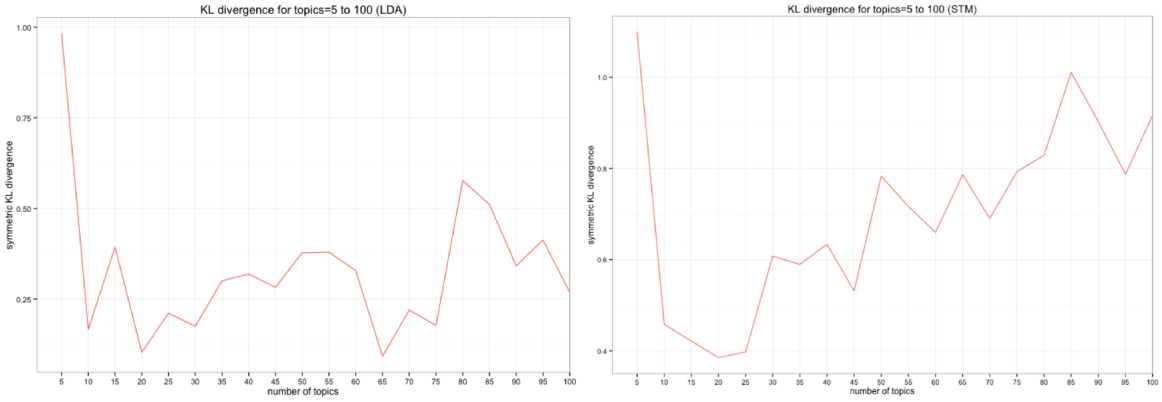

Figure 8: Topic exclusivity as a function of K.

5.2 Model Selection

It is in difficult to compare predictive strength between continuous-response and binary-response models. For this reason, we use several unit-free measures of predictive accuracy: correlation between fitted and observed values, R^2, and distance correlation between fitted and observed values. Correlation canonically measures the strength of linear association between two sequences of random variables, but has a more abstract interpretation as a normalized cosine of the angle between two vectors. Therefore it provides a sense of the "wrongness" of a sequence of predictions (since the angle will be greater if the predictions are worse) that is invariant to location and scale. R^2 here is simply a sign-invariant loss function that is normalized by the sample variance of the response; its interpretation as the "explained fraction of variance" does not apply to an SVM because the fitted values are not conditional means. Finally, distance correlation measures statistical dependence.

For the binary-response models, we also computed the accuracy, sensitivity, and specificity. These do not have an obvious analogue in the continuous case, but among the models for which they are defined they provide an additional sense of prediction performance. Recall that $\text{Sensitivity}(\hat{y}; y) \equiv \frac{\#\ \hat{y}=1 \cap y=1}{\#\ y=1}$. In its most common usage, sensitivity is interpreted as the "true positive" rate, because y typically describes a "yes/no" or "true/false" variable, rather than a "high/low" variable. More generally, it measures the predictive performance specifically on observations for which $y = 1$. In our case, this corresponds to the predictive accuracy on high-salary job postings, or the fraction of high-salary job postings that are correctly classified.

By contrast, accuracy measures the *overall* rate of correct classification, and specificity measures the accuracy for low-salary job postings. We retain the terms "sensitivity" and "specificity" for consistency.

Each topic model was fitted to the entire corpus. Then five-fold cross-validation was used to estimate the prediction performance of each model, and the results were averaged. This simulates a missing data situation, in which the entire corpus is available but its metadata is missing for some observations. The outcomes are reported in Table 2. It is immediately clear that the pure LDA-based models perform at least as poorly as random guessing, except when used to fit an SVM. In fact, the LDA-SVM model has extremely high accuracy on high-salary job postings. This is likely due to the fact that the SVM is discriminative rather than generative, so it is suited to the natural separation in the data that is apparent in Figure 4. The SVM seems to benefit less from adding covariates, or to not benefit at all, compared to the generative models.

A few broader patterns are visible in the table. Adding covariates to the predictive model yields a large improvement for LDA-based models and a modest improvement for STM-based models. Sensitivity is typically much higher than specificity except in the worse-than-guessing LDA models, and typically shows greater improvement than specificity when covariates are added. This, again, is probably due to the fact that *some* of the high-salary postings are far apart from the low-salary postings, but low- and high-salary postings are not visibly separated. The overall impression is that our topic scores are poor predictors for continuous metadata, but are decent predictors for binary metadata.

Model	Predictors	Correlation	R^2	Dist. Corr.	Accuracy	Sensitivity	Specificity
OLS (continuous)	θ_{LDA}	-0.001	0.059	0.077	—	—	—
	covariates & θ_{LDA}	0.679	0.496	0.699	—	—	—
	θ_{STM}	0.590	0.343	0.725	—	—	—
	covariates & θ_{STM}	0.785	0.483	0.774	—	—	—
OLS (binary)	θ_{LDA}	0.006	0.055	0.062	0.498	0.331	0.681
	covariates & θ_{LDA}	0.608	0.363	0.602	0.772	0.868	0.667
	θ_{STM}	0.657	0.430	0.657	0.813	0.819	0.808
	covariates & θ_{STM}	0.683	0.463	0.684	0.820	0.848	0.795
GLM (Bernoulli)	θ_{LDA}	0.007	0.056	0.061	0.498	0.331	0.681
	covariates & θ_{LDA}	0.568	0.440	0.602	0.772	0.901	0.630
	θ_{STM}	0.644	0.499	0.654	0.806	0.855	0.762
	covariates & θ_{STM}	0.643	0.532	0.677	0.824	0.906	0.748
SVM	θ_{LDA}	0.078	0.349	0.101	0.772	0.901	0.630
	covariates & θ_{LDA}	0.553	0.608	0.543	0.765	0.828	0.696
	θ_{STM}	0.698	0.476	0.692	0.824	0.906	0.748
	covariates & θ_{STM}	0.731	0.528	0.727	0.837	0.848	0.828

Table 2: Prediction results for 8 models

5.3 Interpretation

Looking at the summary of the GLM model of θ_{stm} with covariates, we found that the coefficients of topic 2,5,9,13,14, length_text, and residency.bin are statistically significant, and topic 3, 6, 10 and 12 are very close to be significant. Among these variables, topic 6, 12, 13, 14 and length_text are positively related to salary. Taking length_text as an example, for every one unit increase of length of the text of the job, the log odds of the salary of the job being in the upper bin increases by 0.00191. The rest are negatively related to salary. Taking residency_bin as an example, if the job requires city residency, the log odds of the salary of the job being in the upper bin will decrease by 1.437.

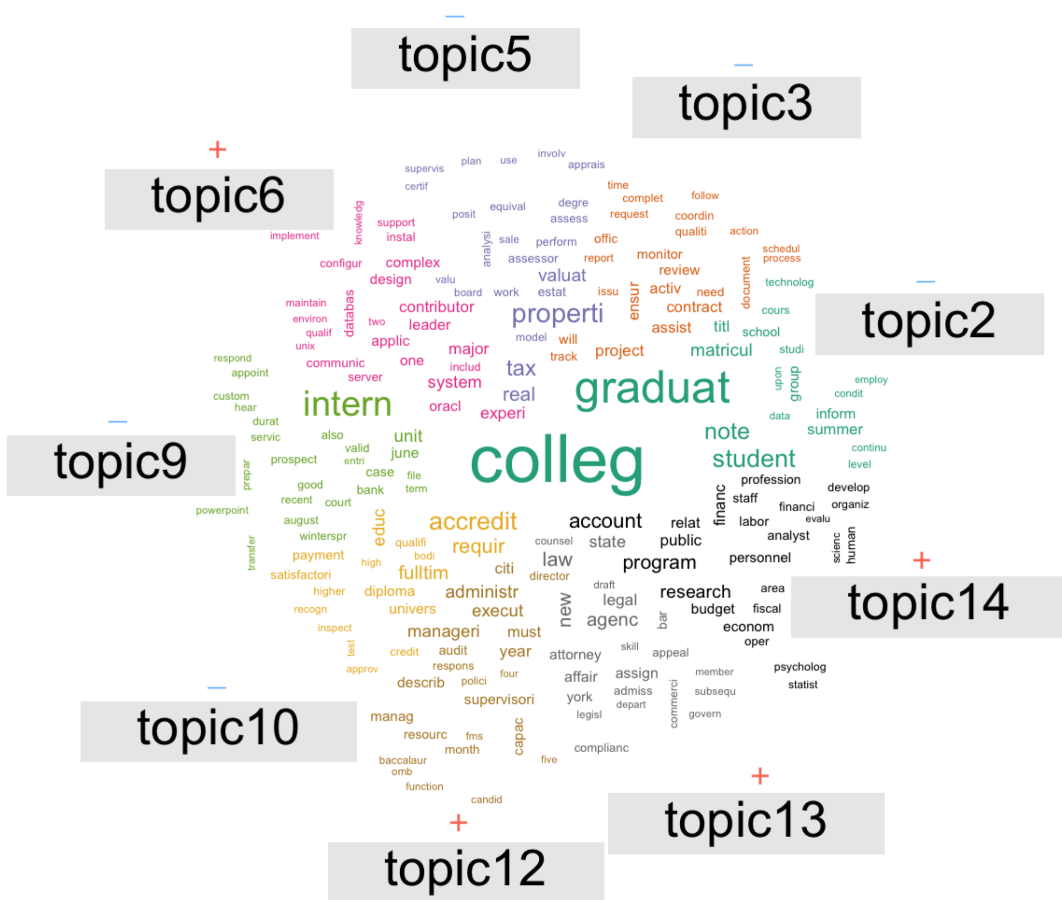

Figure 9: Classical MDS, colored by level and agency.

We plot a cloud comparing word probabilities across all the significant and near-significant topics in Figure 9. In the plot we can see the high-probability words shared by all topics are "colleg" and "graduat", possibly implying that education levels are indicated in most job postings. Topics that are positively related to salary have high-probability words specifying fields of the jobs such as {"server", "oracl", "database", "system"} (IT, *topic* 6), {"manageri", "supervisori", "administr", "execut"} (Managerial, *topic* 12), {"law", "legal", "attorney", "appeal"} (Law, *topic* 13), and {"finance", "analyst", "econom", "statist"} (Finance, *topic* 14), while topics that are negatively related to salary are more likely to be associated with either verbs such as "assist" "follow" "request" "appoint" "respond" and "perform", or words suggesting short terms such as "winterspr" "summer" "intern" and "credit", implying that jobs postings that shared these topics either lean to describe universal primary job functions instead of field-specified ones in their requirements, or are tailored to enrolled students seeking for seasonal internships.

6 Conclusion

We ask whether we can predict the salary offered with a job posting using the text of that posting. Our model has modest performance on salary as a continuous variable, but does well when the target is widened to a binary variable. This will always improve performance by reducing the variance of the target, and thereby leaving less variance to be accounted for by the model. However it also enables the use of a broader class of generative models with efficient implementations in R. It is true that our model assumes that the entire corpus is available and that meta data to be predicted is conditionally missing at random. But this is

not an outlandish scenario, so long as the missingness mechanism can be modeled. Corpora of job postings are widely available; many are only partially labeled, but are also rich in other metadata that could be used to model label missingness. In addition, this technique could be applied directly to other corpora of short, information-rich texts such as Twitter data.

A more general approach would be to fit the topic model only on completely observed documents, then make predictions by imputing topics with the estimated hyperparameters (e.g. γ and Σ_θ) and fitting a model on those imputed topics. No publicly-available software package exists to implement this procedure, so it would have to be done from scratch. This model would be applicable to a much broader class of problems. In addition, the fact that it must be implemented from scratch makes it much easier to write a program to fit the predictive and topic model simultaneously, or to employ full Bayes estimation that utilizes entire distributions of parameters instead of than their posterior modes. Full Bayes modeling also makes it conceptually easier to incorporate shrinkage estimators like the LASSO.

The `stm` package uses variational expectation-maximization to fit the STM model, and it is possible to incorporate the predictive model directly into the EM equations. Moreover, a relatively new kind of Markov Chain Monte Carlo method called Hamiltonian Monte Carlo could be used to implement full posterior and posterior predictive sampling. This could potentially reduce prediction variance dramatically by using distributions of imputed predictors, rather than point estimates of those predictors.

References

[1] David M. Blei. "Probabilistic topic models". In: *Communications of the ACM* (2012).

[2] Brandon M. Stewart Margaret E. Roberts and Dustin Tingley. *stm: R package for structural topic models*. 2014.

[3] C.E. Veni Madhavan R. Arun V. Suresh and M. Narasimha Murty. "On finding the natural number of topics with latent Dirichlet allocation: some observations". In: *Advances in Knowledge Discovery and Data Mining*. 2010.