Advanced Data Analysis Final Project

Team 2:

Duanhong Gao

Yi Jian

Jingwei Li

Aoyuan Liao

Hanqing Shi

Jia Wang

Xiangyu Wu

Yutong Yang (yy2624)

Wanyi Zhang

Fall 2016

**I. Introduction**

The last part of our project aims to use machine learning methods to recommend tags for each question asked on Stack Overflow. We use Latent Dirichlet allocation (LDA) to do topic modeling to all R questions from Stack Overflow, which are described in full detail in the Date Description section. Then for each test question we apply k-nearest neighbors (k-NN) algorithm and ranking of the numbers of tags to recommend tags, finally, to make our result more accurate, we add weights to those tags which appear in the question bodies. The rest of the paper is organized as follows. Section III illustrates the rationale, which is the algorithms we choose and the reasons. Section IV presents the results of recommendation, including the plot of our topics and the most common words in each topic, a sample list of comparison of the actual tags and recommended tags, and the error rate of the whole test data set. Section V draws the conclusion, which states that our method is doing a pretty good job in recommending tags for questions. Also, we suggest possible improvement for the research in the end.

**II. Data Description**

We have two sets of data, one contains questions and answers from Stack Overflow that are tagged with the [r tag](http://stackoverflow.com/questions/tagged/r), another one contains questions and answers from Stack Overflow that are tagged with the [python tag](http://stackoverflow.com/questions/tagged/r).

***‘R Questions from Stack Overflow’*** data set is organized as three tables:

* **Questions** contains the title, body, creation date, score, and owner ID for each R question.
* **Answers** contains the body, creation date, score, and owner ID for each of the answers to these questions. The ParentId column links back to the Questions table.
* **Tags** contains the tags on each question *besides* the R tag.

***‘Python Questions from Stack Overflow’*** data set is also organized as three tables:

* **Questions** contains the title, body, creation date, score, and owner ID for each Python question.
* **Answers** contains the body, creation date, score, and owner ID for each of the answers to these questions. The ParentId column links back to the Questions table.
* **Tags** contains the tags on each question *besides* the Python tag.

For space reasons only non-deleted and non-closed content are included in the dataset. Both datasets contain questions up to **19 October 2016** (UTC).

(data description for first two parts of this project)

In the last part of our project, we only use ***‘R Questions from Stack Overflow’*** data setto do tag recommendation, since the methodology is the same for different data sets. And we only use ‘**Questions**’ table and ‘**Tags**’ table in this data set. For ‘**Tags**’ table, we delete tags which show up less than five times during the whole time period. We randomly split ‘**Questions**’ table into training (80%) and testing (20%) sets.

**III. Methodology**

In order to recommend tags for questions, we need to find the most similar questions in training set, and choose the most common tags as recommendations. For each test question, we first do LDA topic modeling to cluster questions into 50 topics, and then use k-NN to find the nearest 20 questions to the test question, and rank the tags belonging to these questions. After that, we choose 20 top tags, and look through the body of the test question, add weights to tags which show up in the question body. Finally, we rank these 20 tags again after weighting and choose the most likely 6 tags as recommendations. More specifically, we use the following steps to do recommendation.

To begin with, we deal with our data as described in Section II to obtain the useful data for our algorithms. After this, we have a training data set and a test data set of questions.

Secondly, we apply LDA to the whole question data set, and cluster these questions into 50 topics.

Thirdly, to find the nearest questions for each test question, we use k-NN algorithm to all training questions belonging to the same topic as that test question with a distance defined according to Jaccard index, which is the size of intersection divided by the size of the union of unique words in test question body and unique words in training set question bodies. By doing this, we obtain 20 nearest questions to each test question.

Finally, we rank the tags belonging to the 20 chosen questions, and pick up the top 20 tags as choices. To improve our prediction accuracy, we compare these 20 tags with the body of the test question, add weights to each tag which appears in question body because they are more likely to be chosen as tags. Then we rank 20 weighted tags again, and keep the top 6 as recommendations for customers to choose from.

As for the accuracy of our recommendation, we calculate an error rate using a measurement that our recommendation is right as long as one of the actual tag appears in recommended tags.

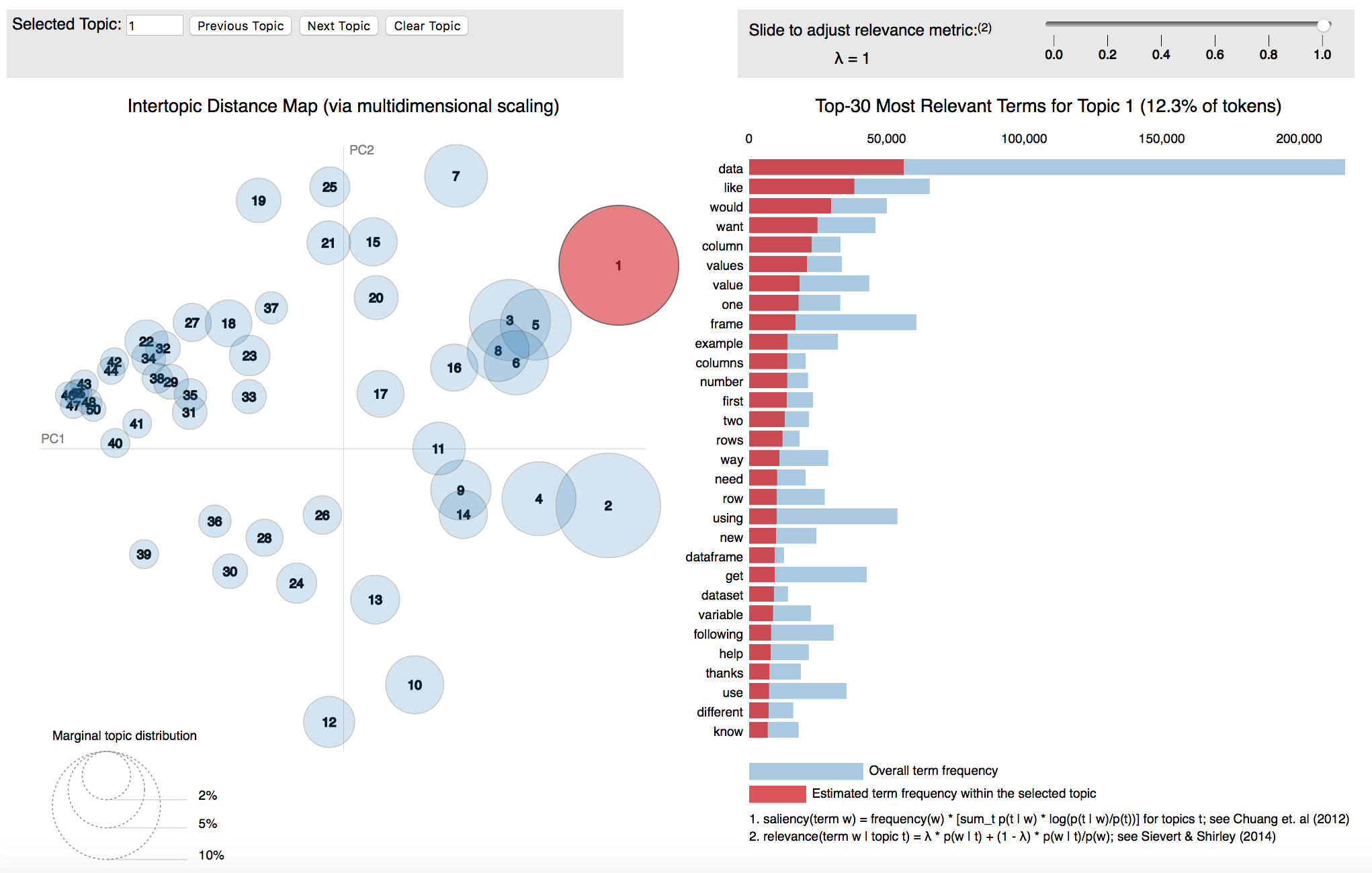
**IV. Empirical Results**

We use a visualization tool (Figure 1) to visualize our topic model. To use it, click a circle in the left panel to select a topic, and the bar chart in the right panel will display the 30 most relevant terms for the selected topic, where the definition of relevance of a term to a topic is ,for a given weight parameter, .

The red bars represent the frequency of a term in the chosen topic (proportional to), and the blue bars represent a term's frequency across the entire corpus (proportional to ). The area of the circles depends on each topic’s overall prevalence.

Change the value of λ to adjust the term rankings -- small values of λ (near 0) highlight potentially rare, but exclusive terms for the selected topic, and large values of λ (near 1) highlight frequent, but not necessarily exclusive, terms for the selected topic.

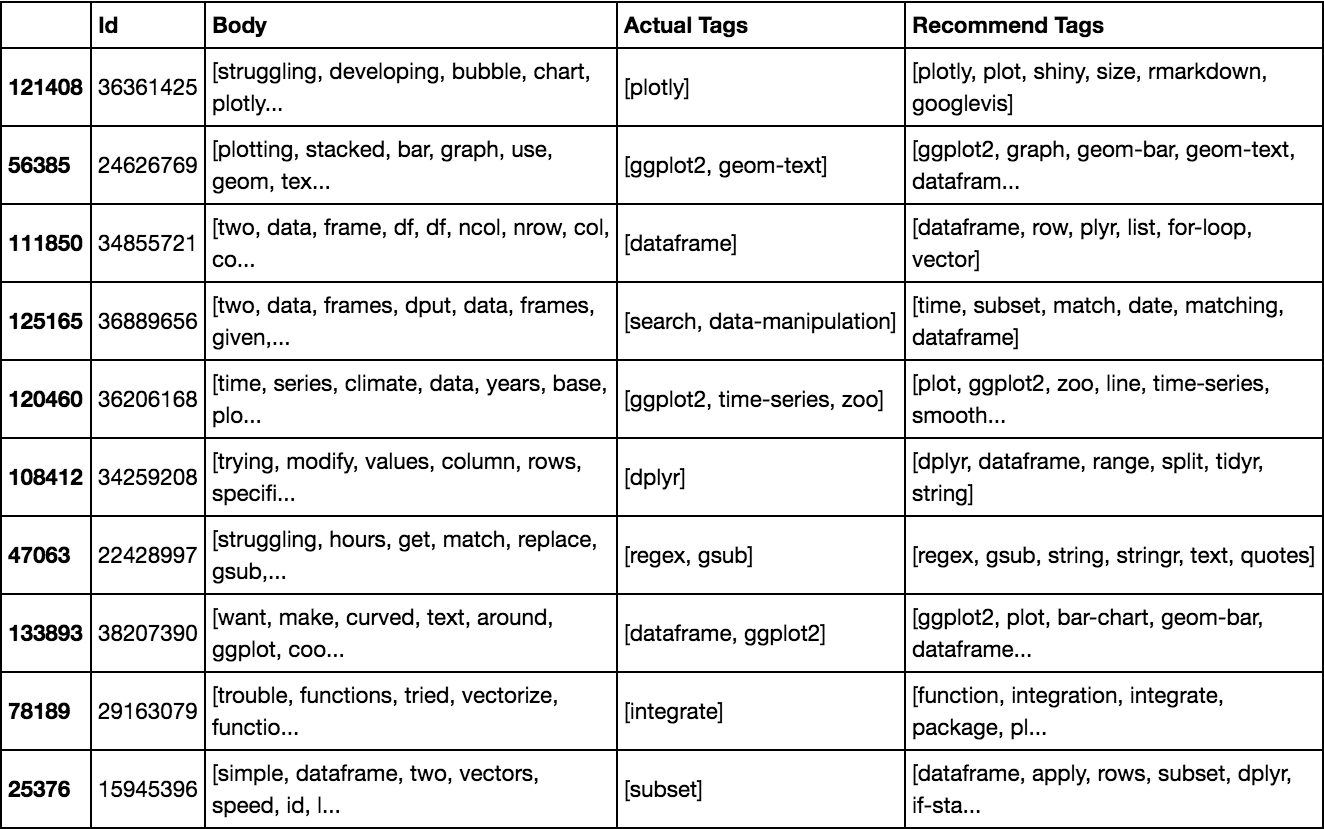
*Figure 1: visualization and interpretation of topics*



Remark: The layout of LDAvis, with the global topics (50 topics in our case) view on the left, and the term barcharts (with topic 1 selected) on the right. Linked selections allow users to reveal aspects of the topic-term relationships compactly.

We run our algorithms and predict tags for the test set. There are some sample of our recommendation results listed in table 1.

*Table 1: A sample of comparison between actual tags and recommended tags*



After prediction, we calculate an error rate as a measure of goodness of our model and the error rate is calculated using method described in Section III. The error rate is 29.21%, which is a pretty low error rate, since in many recommendations, although we don’t recommend the exact same tags as the actual tags but the recommended tags are strongly related to the questions and actual tags.

**V. Conclusion**

In short, we use LDA, k-NN algorithms and a weighting technique to do tags recommendation, which is very useful in real world. Now, all tags for questions on Stack Overflow are added by customers themselves, Stack Overflow only recommends tags after customers input some letters of the tags they are adding. But what we do in this project can help add tags on questions automatically after the questions are finished. It’s very helpful for all customers, especially for customers who don’t know how to add tags very well. And our recommendations are usually right and suitable due to the low error rate.

Further research should be directed towards the application of our model, and improving the accuracy of our recommendation by correcting details in our algorithms, such as, improving our weighting technique and learning more about the objects which we add weights on.