Data Mining Specialization Capstone Project. Final Report

Juan Luis Herrera Cortijo (juan.luis.herrera.cortijo@gmail.com (mailto:juan.luis.herrera.cortijo@gmail.com))

1. Introduction

Today, customer reviews in social media have a deep impact on the chances of success of any business [1]. Restaurant customers look for a complete and satisfactory experience regarding food quality, service, and ambient and they often seek the opinion of patrons when they are choosing a place for their next meal. Yelp offers this information to its users. When users look for a place to eat, they can ask the service for a list of close restaurants that fall in some cuisine category. Users also get the overall rating that other customers gave to the restaurant as well as some reviews about the restaurant.

Reviews content is very diverse. They can talk about the food, the service, the ambiance; they can reflect a positive experience or be a complain about some specific aspect of their experience. Therefore, reviews are a wealth of information and usually are more informative than a numeric rating. On the other hand, a service like Yelp receives thousands of reviews each day from every corner of the world and summarizing or extracting specific pieces of information from such a big corpus is a challenging task.

Data mining and more concretely text mining techniques allow us to explore a massive corpus like the one of Yelp reviews. We can obtain new insights about the text content that may be helpful for customers, restaurant owners, government or even for Yelp.

In this project, we mine a corpus of Yelp restaurant reviews to explore the next questions: What topics are frequently treated in the reviews? How are different cuisines related? Can we recommend dishes for a particular cuisine and which restaurant is best to try them? Can we predict whether a restaurant will pass its hygiene inspection?

For reproducibility. All the R code used to generate the results in this report can be found in this GitHub repository (https://github.com/Belethia/Topic-Mining)

2 The data

The data used in this project is part of the Yelp Dataset Challenge (http://www.yelp.com/dataset_challenge).The dataset consists of a set of JSON files that include business information, reviews, tips (shorter reviews), user information and check-ins. Business objects list name, location, opening hours, category, average star rating, the number of reviews about the business and a series of attributes like noise level or reservations policy. Review objects list a star rating, the review text, the review date, and the number of votes that the review has received.In this project, we have focused on these two type of objects.

We filtered the business by category to keep only those businesses in the restaurant category (14303) and reviews related to those businesses (706646).

The texts from restaurant reviews will form the basic corpus of this project.

2.1 Language model

Many of the techniques applied require to use a document-term matrix as input. To obtain such matrix we have processed each of the reviews to build a bag of words language model. To create this model we preprocessed each document in the corpus as follows:

- Remove non-writable characters.
- Strip extra white spaces.
- Lower case.
- Remove punctuation
- Remove numbers
- Stemming
- Stop words removal.

After that, each text was tokenized into unigrams, and the unigram frequencies were counted and stored into a document-term matrix of counts.

Term counts across all the corpus showed a typical Zipf distribution. Unless otherwise stated, we kept the most frequent terms that, summing all their frequencies, accounted for about 99% of the total number of words in the corpus. The resulting vocabulary has 15697 words.

3. Summary of Specific Tasks

3.1 Task 1. What topics are frequently treated in the reviews?

The first question that could come to our minds is: what are the reviews talking about? Learning which topics are the most frequent among customer reviews and how they associate to a positive or negative rating can help business improve their offer and have a better chance of succeeding. Therefore, our first task was to explore which topics are frequent in the reviews.

Topic model

To discover latent themes in our corpus, we run a Latent Dirichlet Allocation [2] algorithm (LDA) using the document-term frequencies matrix of our language model as input. To estimate the model parameters we used a Gibbs [3][4] sampling with a burn-in phase of 1000 iterations and later the distribution was sampled every 100 iterations during 2000 iterations. We tested other approaches (LDA with VME parameter estimation and a Correlated Topics Model [5]) but the topics obtained were less clear than the ones resulting from LDA with Gibbs sampling.

Results

To get an overall idea of the topics treated in the reviews, first we used all the restaurants as input for a 20 topics model. Most of the 20 topics obtained are well defined. Figure 1 shows them and the four most frequent words for each topic. Color luminance shows the relative in-topic relevance of each term, being the most bright the most common word. There is only one topic that we could not identify.

There are several topics about the customer experience: Disappointed, Love, Returning, Waiting, Good service, Nice, Take Away and having a Special Dinner.

The remaining topics focus on the type of restaurant, according to their cuisine type: American, Asian, Mexican, Italian, Sushi; or other features like Buffets, Location, and Ambient.

There is a “No restaurant” topic because, although all the reviews are for businesses labeled as restaurants, some of them are not (for example, hotels). Also, the “Nightlife” topic focuses on clubs, music venues and other businesses related to night entertainment, but that are not restaurants.

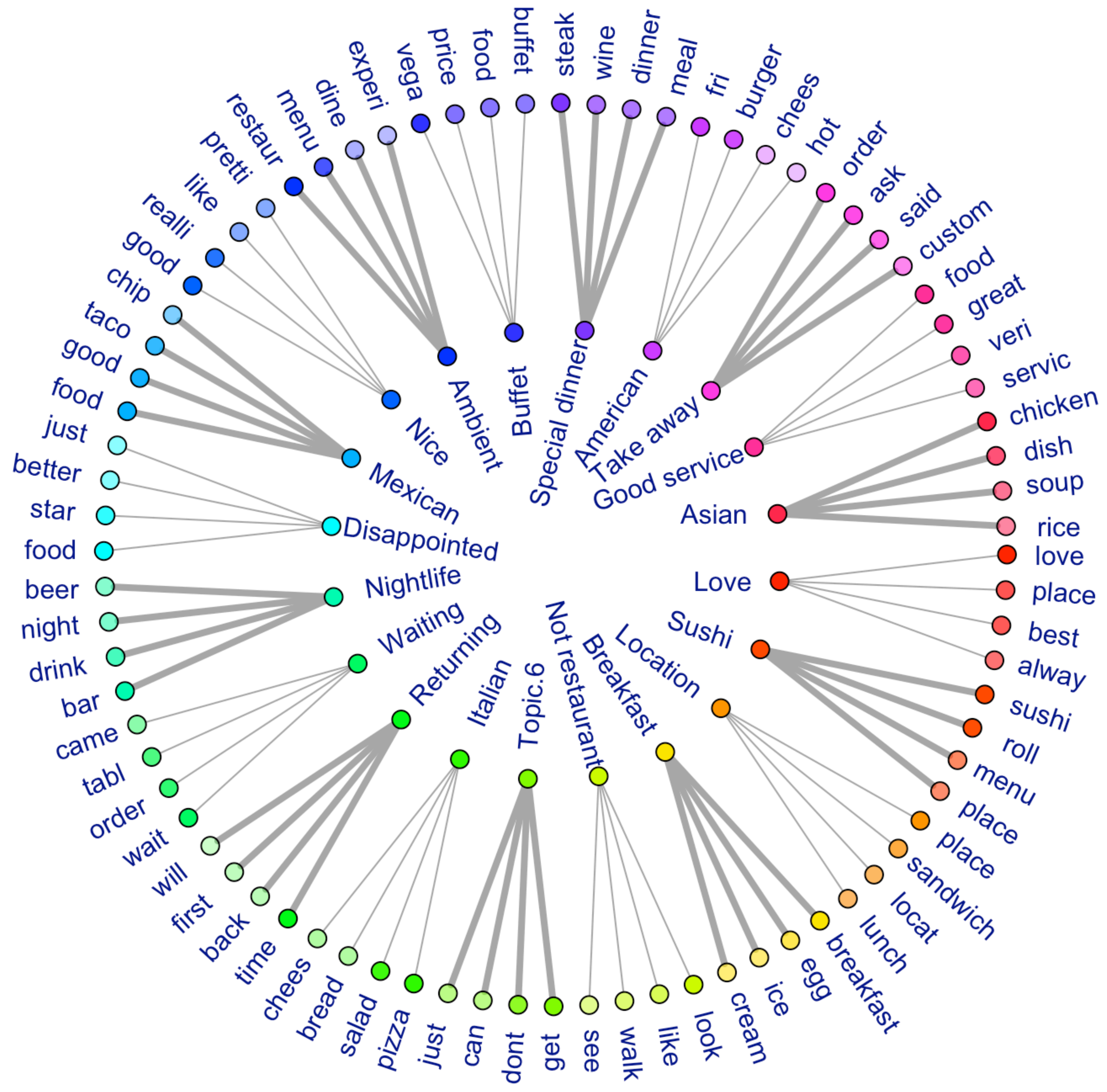

Figure 1. A topic model for Yelp restaurant reviews. Only the four most frequent words are shown for each topic. Color luminance shows the relative in-topic relevance of each term, being the most bright the most frequent term.

The overall topic model lists several topics about customer experience. However, good and bad experiences are mixed because the corpus includes both positive and negative reviews. We have explored the topics related to positive and negative ratings independently. Figure 2 displays the restaurant review rating distribution. We can see that positive reviews (stars >3) dominate over negative reviews (stars <3).

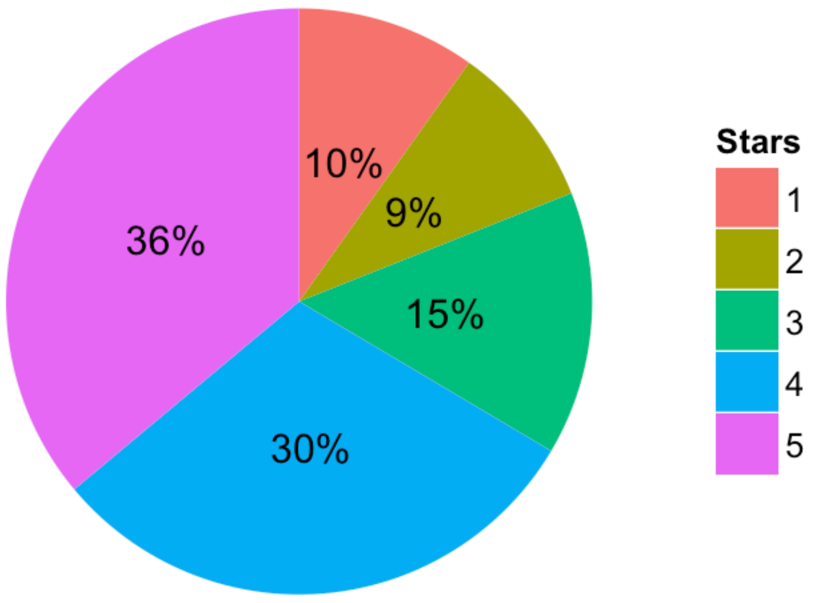

Figure 2. Restaurant review rating distribution.

We have fitted two topic models (one for positive reviews and another for negative reviews) with 20 topics each and following the same methodology that we used to compute the overall topic model. We did not include reviews with three stars because that rating is not positive nor negative.

In general, we can see that some of the overall topics also appear in these two new models. Nevertheless, we get a finer grain topic distribution about the customer experience.

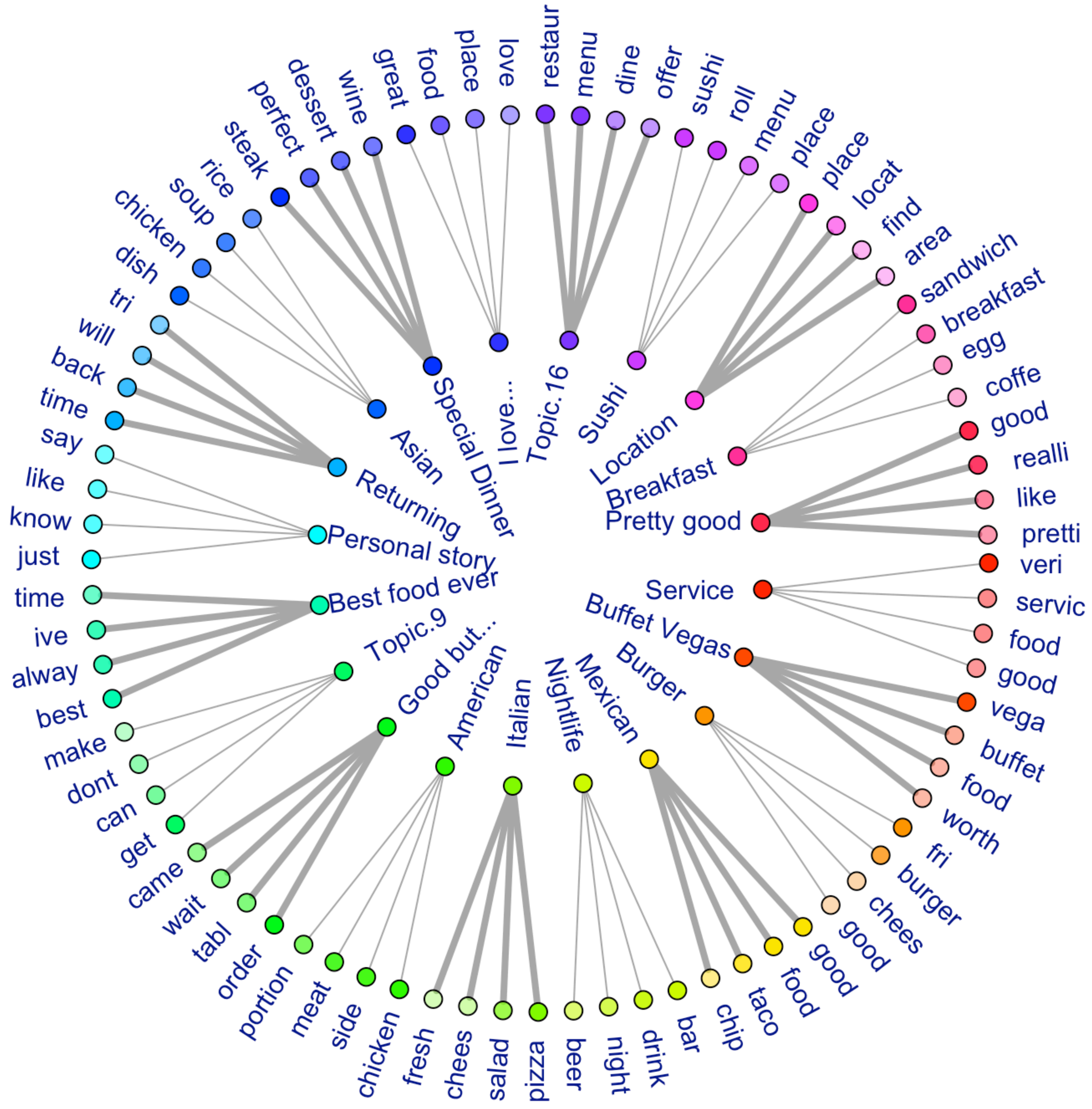

Figure 4. A topic model for negative Yelp restaurant reviews (stars <3). Only the 4 most frequent words are shown for each topic. Color luminance shows the relative in-topic relevance of each term, being the most bright the most frequent term.

3.2 Task 2. How are different cuisines related?

The restaurants listed in the Yelp dataset, are often labeled according to the kind of cuisine that they serve. How those categories are related is an interesting question that allows to draw a map of cuisines.

To investigate this question we have used a subset of the corpus and language model detailed above. An exam of positive restaurant reviews (stars >3) shows that they detail more aspects of the restaurant cuisine than negative reviews. Therefore, we will use only positive reviews in this analysis.

Businesses are classified according to their categories field. Each business can have multiple labels and, in addition to the restaurant label, restaurants receive other tags to specify the type of restaurant. Some of those labels correspond to easily recognizable types of cuisines such as American, Italian and Chinese. Others tags refer to businesses that may or may not serve food (Karaokes or Swimming Pools for example) but that do not have a particular type of cuisine associated. A third kind of label, like Taxi or Automotive, lack a definite relationship with food. To have a good training set to create a cuisine map I have used only the reviews related to business that has one of the labels that clearly points to a type of cuisine.

Finally, Yelp users can vote other user's reviews to highlight that they are useful and accurate. We expect that upvoted reviews will contain better descriptions of the restaurants and their cuisine, and we have used only reviews upvoted at least one time.

The resulting subset includes 205156 reviews of 126 restaurant categories. We aggregated all the reviews by cuisine type by concatenating all the reviews of each category and creating a document for each cuisine. This resulted in 126 long documents. The vocabulary includes 15695 words.

To discover the relations among the cuisines, we have used a document-term matrix to compute the similarities among the reviews of each cuisine type pair. Then we have used the similarities to create a graph representation of cuisines relationships and applied community detection to cluster the types of cuisines.

We have tested three combinations of transformations of a document-term matrix and similarity functions: no transformation + cosine, BM25+[6], and LDA Topic model weights + cosine. The third approach gave the best results and is the one that we describe here. We fitted a topic model to the raw frequencies document-term matrix using LDA as described above. Then we used each document's probability distribution over the topics as a vector representation instead of raw counts and computed cosine similarity among all the cuisines. The result is a 126x126 similarity matrix.

We fitted a 20 topics model, but we only used 19 of them to compute the similarity matrix. The topic discarded doesn't describe any cuisine aspect but the overall positive customer experience common to all the reviews in this subset. Figure 5 shows the similarity matrix computed using a topic model and cosine similarity. Darker cells indicate a stronger similarity between two given cuisines. The rows and columns have been reordered according to the output of a complete linkage hierarchical clustering algorithm. This way, closely related cuisines depict darker triangles close to the matrix diagonal. Off-diagonal, darker cells indicate cuisines that act as connections between clusters. We can see some groups, like the Asian cuisines, that reflect the geographic proximity of their origin countries while others like the one includen Bakeries reflect a general type of food like sweet food.

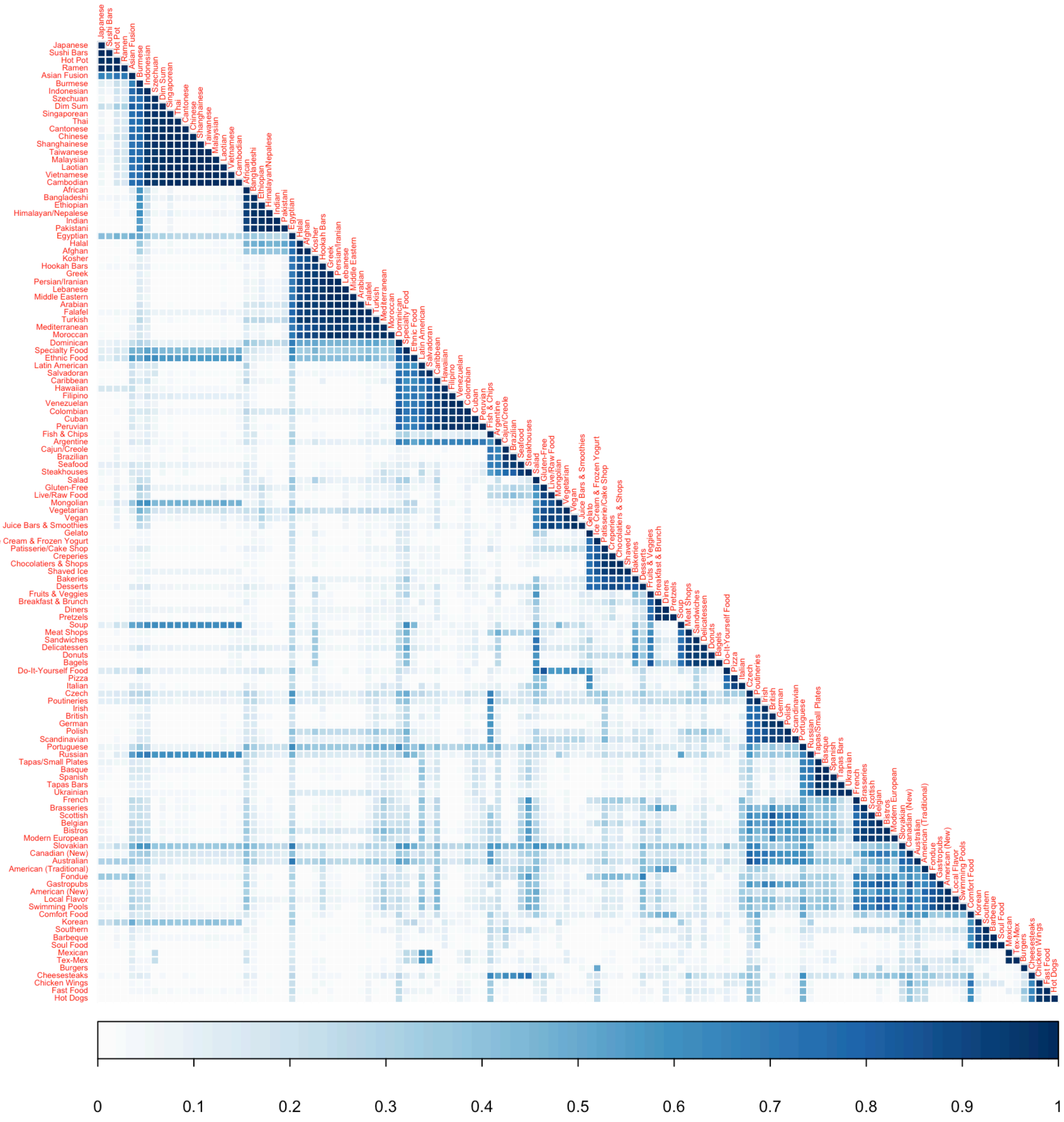

Figure 5. Similarity matrix computed applying cosine similarity to the document weights of a topic model. Rows and columns have been reordered according to the output of a complete linkage hierarchical clustering algorithm

Although this representation clearly shows some groups, it makes hard to infer the relationships between groups. Each cell in the cuisine similarity matrix represents the strength of the relationship between two cuisines. We can represent the similarity matrix as a fully connected graph in which the nodes are the cuisines, the edges the relationship between cuisines and the similarities are the edges weight. To get a representation of the most relevant relationships among cuisines we need to transform that fully connected graph into a sparser graph that only retains essential relationships (a backbone graph).

We have followed [7] to compute a backbone graph for each similarity matrix. We started using a confidence level of 0.001 and increased the confidence level in 0.001 steps until we got a connected graph that included all the cuisines, or the confidence level reached a value of 0.05. To achieve this step we used the implementation in the R package `disparityfilter`.

To cluster the cuisines in the map, we have tested all the community detection algorithms listed in the R igraph package. The one that performed better and we applied on the cuisine maps is the one based on label propagation [8].

Figure 6 shows the backbone network computed from the similarity matrix shown in Figure 5. Colors indicate clusters obtained using the community detection algorithm. The backbone graph includes all the cuisines, and the significance level of the edges is less or equal to 0.011. As expected from the similarity matrix, we can observe several clusters that describe meaningful relations among cuisines. For example, Asian cuisines group in a well-defined cluster, the same happens with Arabic, South American, and many European cuisines. However, geographic location is not the only criterion that define the clusters found, types of dishes like desserts also appear clustered. In addition to tight clusters, we can find some cuisines like Australian, Argentine or Fondue that work as a bridge between clusters. These bridges do not always have a culinary meaning. Some of the bridge cuisines are those with fewer reviews, and their relationships may be caused by spurious similarities in the review contents.

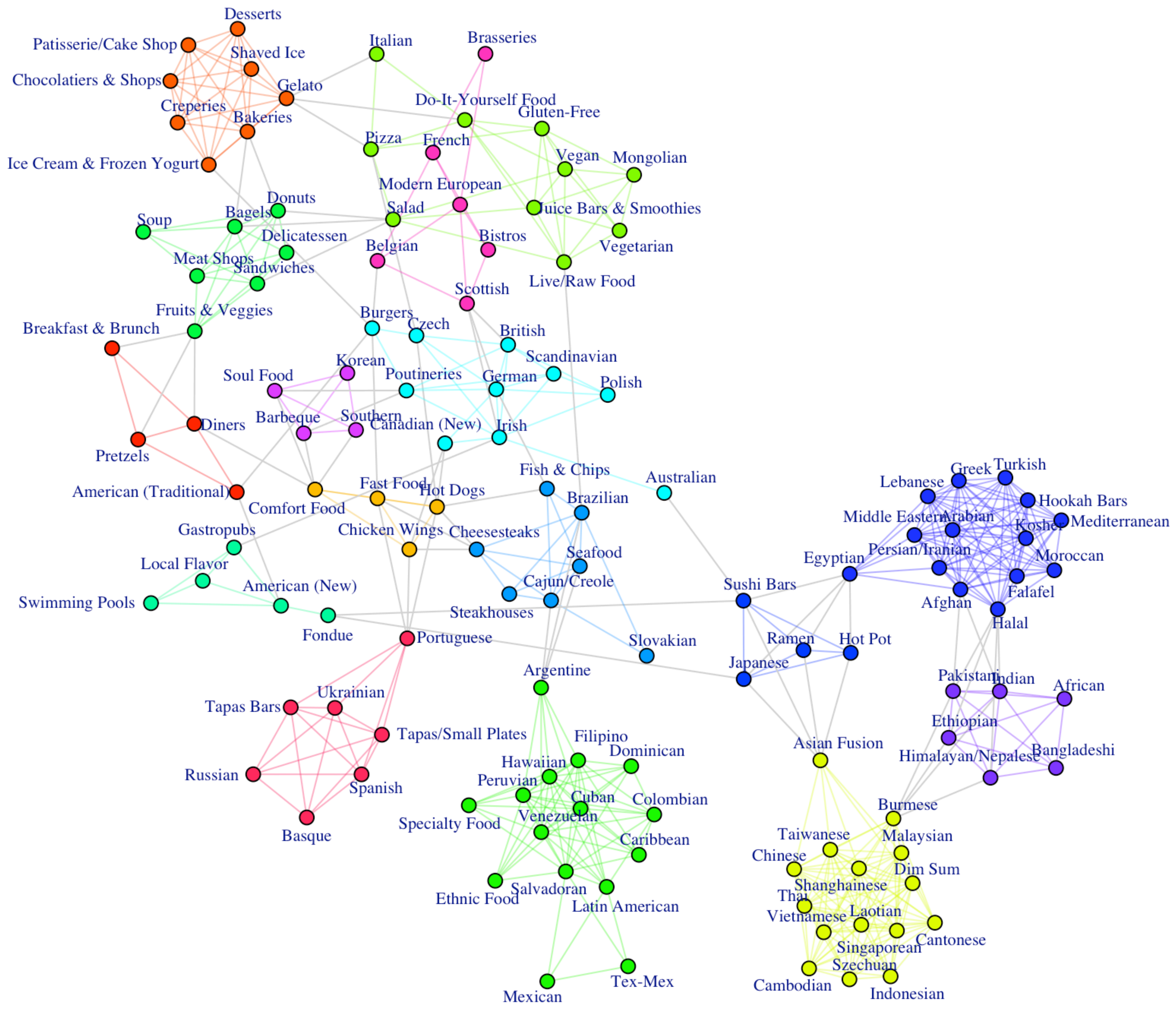

Figure 6. Cuisine map obtained by extracting a backbone graph from the similarity matrix displayed in Figure 5. Colors indicate clusters defined using a label propagation community detection algorithm.

3.3 Tasks 3,4 and 5. Can we recommend dishes for a particular cuisine and which restaurant is best to try them?

In tasks 3,4 and 5 we focused on extracting new, specific knowledge from the reviews and creating a product that could be useful to the users, the restaurant owners or even Yelp. In particular, we extracted dish names from the reviews and created a ranking of popular dishes for a given cuisine that we could offer to a user as a guide to discovering new dishes. Also, we built a recommender system of restaurants that cook a dish.

3.3.1 Dish extraction

In this step, our objective is to extract dish names from a corpus of Yelp reviews. In particular, we try to compile a list of dishes for six cuisines: New American, Chinese, Indian, Italian, Mediterranean, and Mexican. For that, we have applied three different algorithms for phrase extraction: SegPhrase [1], TopMine [2] and word2vect [3].

TopMine

TopMine (http://web.engr.illinois.edu/~elkishk2/) first segments each review into single and multi-word phrases and then applies an LDA topic model on the partitioned document. The algorithm estimates the topic model hyperparameters using Gibbs sampling. The output of the algorithm includes the distribution by topic of single and multi-word phrases.

We run the algorithm for each cuisine using the following configuration.

- minimum phrase frequency: 5
- maximum size of phrase (number of words): 6
- number of topics: 10
- Giggs sampling iterations: 500
- significance threshold for merging unigrams into phrases: 4
- burnin before hyperparameter optimization: 100
- alpha hyperparameter: 2
- optimize hyperparameters every n iterations: 50

We selected the number of sampling iterations inspecting the evolution of perplexity during the optimization and selecting the point in which the perplexity reduction after 50 steps was not significative compared to previous iterations.

After running the algorithm, we inspected the resulting topics, and we selected those topics clearly related to dishes and food. Of all the terms chosen for one cuisine, we only kept those with a frequency above a threshold determined by the quality of the items above and bellow the threshold.

SegPhrase

This algorithm extracts popular phrases by first segmenting the corpus and selecting those phrases above some minimum threshold support and then rectifying the phrase counts according to phrase quality criterias. The algorithm allows as input a set of labeled good and bad quality phrases that allows to rectify the counts to extract phrases closely related to the positively labeled samples. Here we used a list of items obtained using a feature of SegPhrase that allows get a list of phrases autolabeled. We reviewed manually the autogenerated list to remove false positives and negatives. After that, we expanded the list by including dishes listed on Wikipedia.

We cloned the SegPhrase repository on GitHub (https://github.com/shangjingbo1226/SegPhrase) and applied the tool using default parameters except for:

- support threshold: 10.
- enabled use of unigrams.
- enabled use of word networks.

After that, we filtered the phrases listed in the file salient.csv to keep only those phrases with a rectified count above 0.8.

Word2Vect

The word2vect (https://code.google.com/p/word2vec/) trains a neural network to compute a vector representation of words focused on the word similarity task.

First we obtained a word representation that included phrases with two and three words by running twice the word2phrase tool provided along with word2net.

```
word2phrase -train corpus.txt -output corpus_w_bigrams.txt
word2phrase -train corpus_w_bigrams.txt -output corpus_w_trigrams.txt
```

Then, we computed the vector representations using the word2vect program

```
word2vec -train corpus_w_trigrams.txt -output model.txt -binary 0
```

the option binary 0 writes the program output in a CSV file.

Once we had the vector representation, we loaded the output into R, removed the vectors for stopwords (in English and any other relevant language) and computed the cosine similarity among all the words.

Finally, to get a list of dishes for each cuisine, we used some dishes names as a starting point and searched for similar words using the similarity matrix. We used a similarity threshold between 0.7 and 0.8, depending on the cuisine. As a starting point, we used the positively labeled dishes in the labeled list described in the SegPhrase algorithm.

Results

All the methods applied retrieved valid dish names and, although the highest scored results were dishes, the quality of the results deteriorated rapidly as the score of the items was lower. Therefore, to keep only good quality results, we inspected the output and fixed the thresholds described above. We discarded all items with scored bellow the corresponding threshold.

Proceeding this way, we got one list of items for each algorithm and each cuisine. Word2Vect resulted in a list containing more items than the other two algorithms.

For brebity, we only show some results for Italian cuisine and TopMine. Figure 7 display the items retrieved by TopMine.

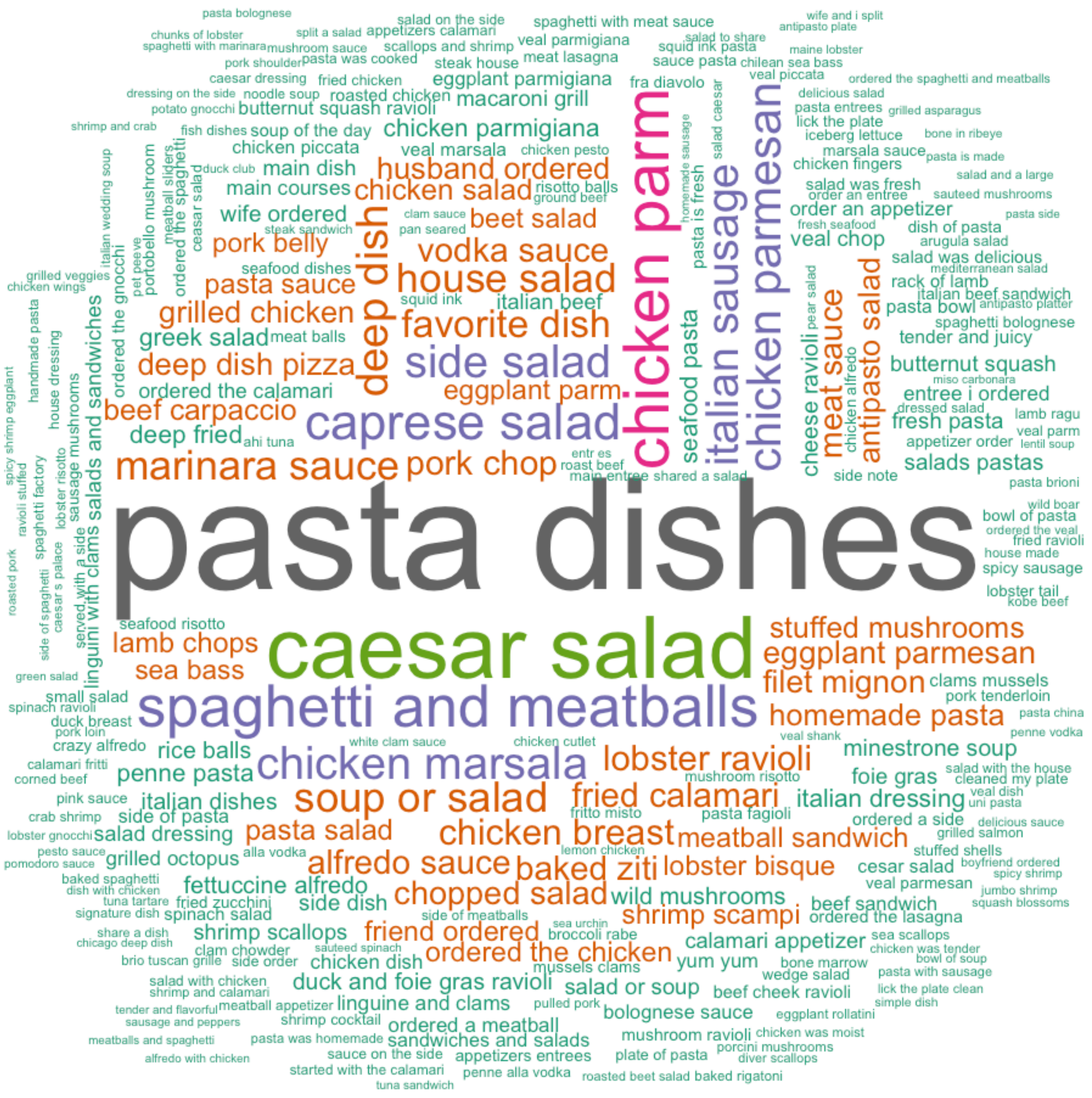

Figure 7. Wordcloud displaying the list of dishes obtained applying TopMine to the Italian corpus. The size of the words is related to the item frequency given by the algorithm.

However, even after keeping only the top scoring items for each algorithm we found items not related to food, related to food but that were not dishes (for example, ingredients), and dishes from other cuisines.

To further refine our results, for each cuisine we pooled together all the items obtained using the three algorithms, removing duplicates. Then we followed two approaches:

1. Determine which items appear in more than three cuisines and remove them from all the cuisines. If one item is listed in more than three cuisines, there is a good chance that it is not specific of any cuisine. We removed such common items from all the cuisines lists.

2. Some cuisines share items with other cuisines because they receive a strong influence from others (for example the New American cuisine). In this case, we removed from the influenced cuisine the items that it has in common with the influencing cuisine.

We have used the resulting lists of dishes in the next two steps.

3.3.2 Dish Ranking

A dish ranking is a useful tool for those who want to explore a cuisine because it tells them which dishes are the most popular and, therefore, the most representative. To create a dish ranking, we need some measure or score to compare the dishes according to their popularity. Once we have such score, we simply list the dishes in decreasing order of the score, and we present to the user the top-N dishes. We have computed four different scores: document frequency, average review stars, and versions of the two scores weighted by the probability that the reviews belong to a food topic.

Document frequency

For each dish, we counted the number of reviews that contain the dish name. We used the document frequency as opposed to the term frequency because some reviews mention a dish repeatedly, and this would result in an artificial inflation of popularity.

Some dishes are not exclusively served by restaurants of one category, especially if the two cuisines are closely related, like Mediterranean and Italian. So, we did not restrict ourselves to count the number of occurrences in the reviews related to the corresponding cuisines category.

Average review stars

For each dish, we have computed the average stars of the reviews that mention it. As in document frequency, we have used all the reviews as opposed to restricting to the corresponding cuisine category.

Food topic probability

We have computed a topic model using LDA as explained in above. In this case, we have used all the reviews and just three topics. Then we inspected the three topics and selected the topic related to food. The other two topics were about service and ambiance.

Our score is a modified version of the term frequency score. Instead of summing 1 for each document that mentions a dish, we sum the probability of the review being assigned to the food topic. So, our score is a weighted document frequency that highlights reviews focused on the culinary experience.

Average review stars weighted by food topic probability

Our last approach combines each review rating and the probability of being assigned to the food topic. For one dish, the average score includes the opinion of the customers about their experience and how related to food are the review in which the dish appears.

This score reduces the impact of high scores earned by other aspects of the experience rather than by the food. For example, a customer might be very happy with the service and the dish be mentioned only briefly and not be the reason for the rating.

Results

Although we have computed the ranking for all the dishes in the lists of dishes obtained in the pervious step, for readability we only will show the top-20 dishes in each ranking.

Again, for brevity we will only show the results for Italian and Chinese cuisine. Figures 8 to 11 show dish rankings for the two cuisines computed using each of the scores described above. The first thing that we can notice is that some of the dishes are not dishes (Figure 8). The reason is that we obtained the list of dishes using an automatic mining procedure that introduced some false positive dishes in our lists. This problem seems to be reduced partially by using the average review rating to rank the dishes (Figure 9).

Using weighted document frequency (Figure 10) moves down some positions those words not related to food. The best result regarding keeping only dishes in the ranking is obtained using the weighted stars average (Figure 11).

When we compare Figures 9 and 11, we can see that the dishes do not follow the same order. For example, in the Chinese cuisine we can see that “aji panca” moves from lower positions in Figure 9 to the first position in Figure 11. On the other hand, dishes like “taro snoh” disappear from the top-20 ranking. A closer exam of the reviews that mention both dishes shows the reason. Almost all the 15 reviews that mention “aji panca” are a detailed description of what customers ordered and their opinion about the dishes. On the other hand, most of the five reviews that mention “taro nosh” talk about something else like the ambiance or the service. When we weighted the ratings using a topic model we reinforced opinions focused on culinary aspects.

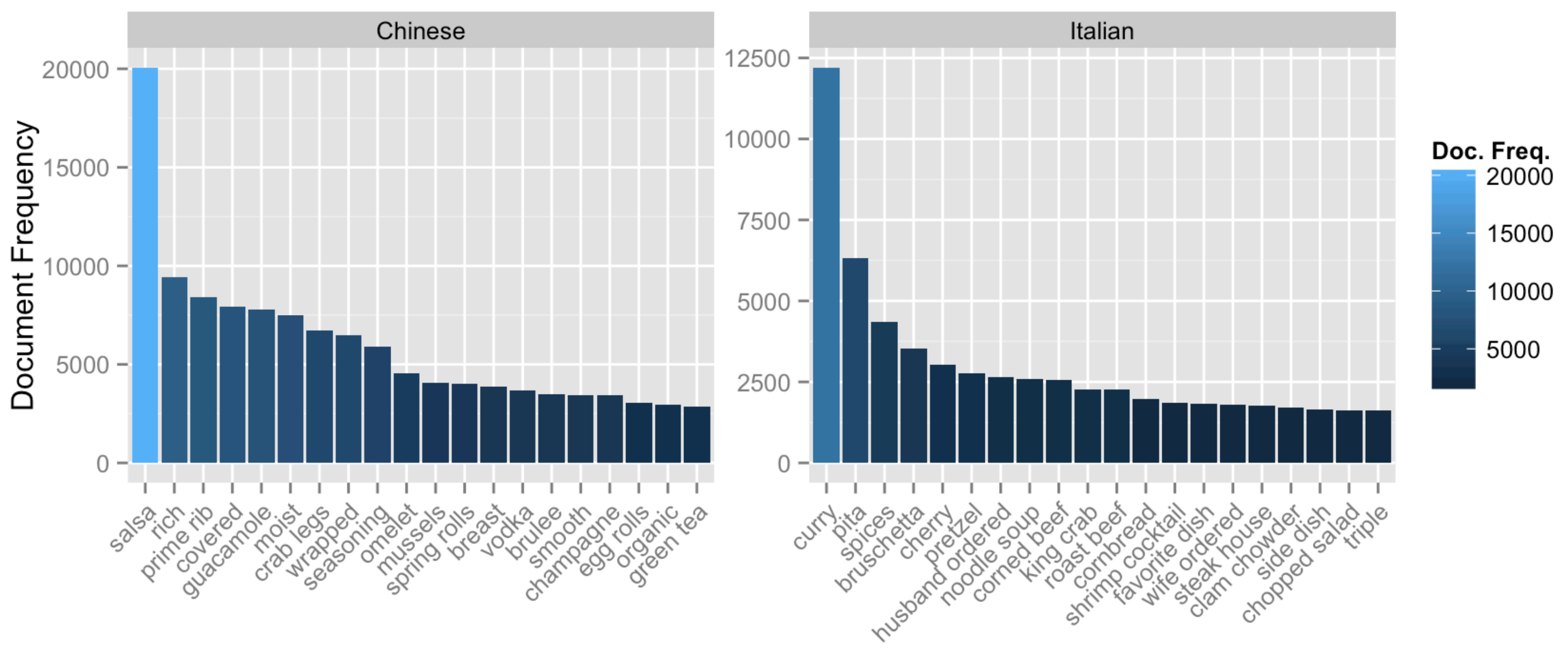

Figure 8. Dishes ranked by document frequency for two cuisines.

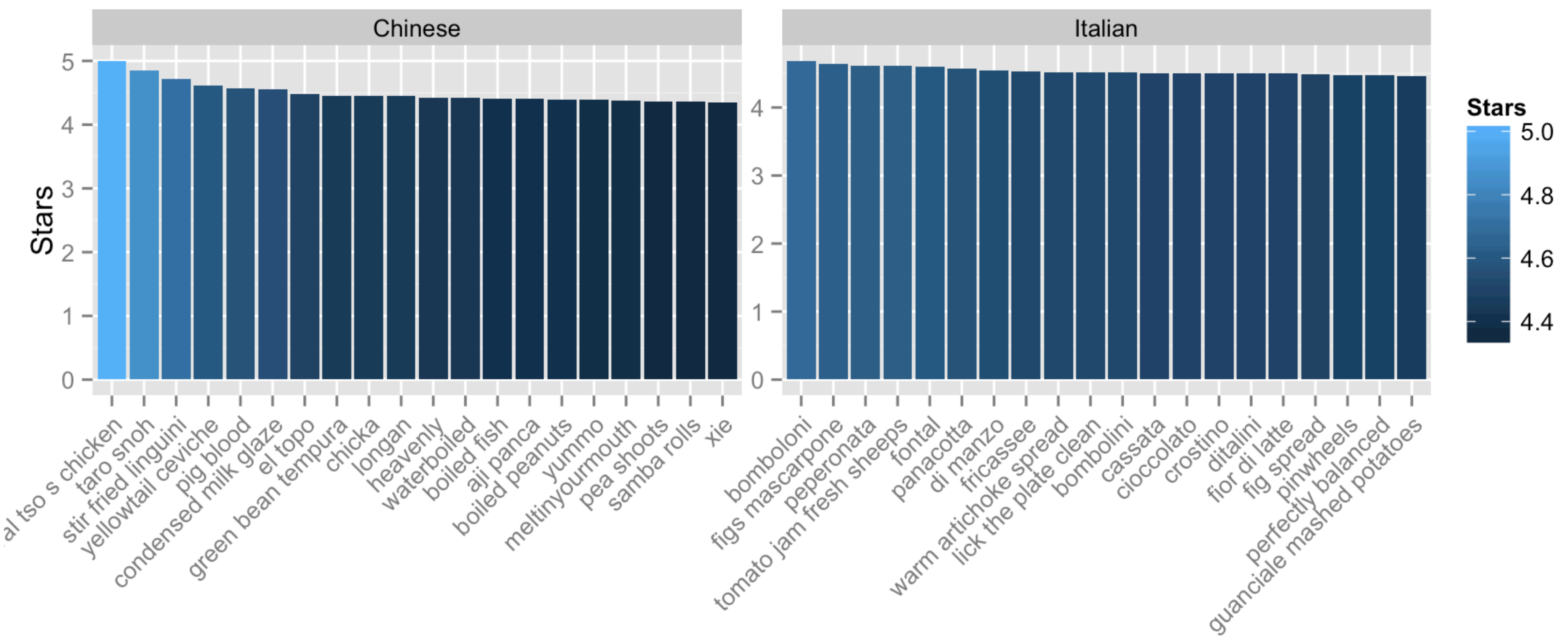

Figure 9. Dishes ranked by average review stars for two cuisines.

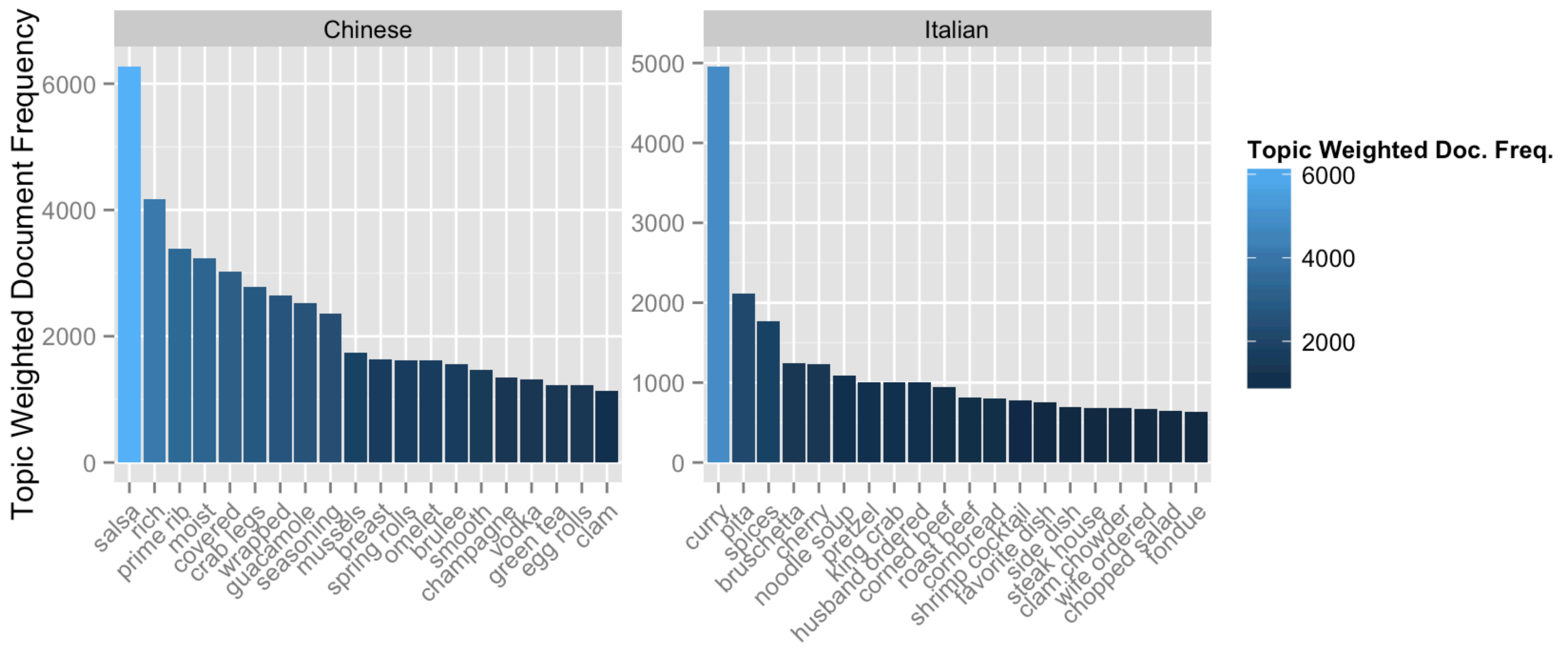

Figure 10. Dishes ranked by topic weighted document frequency for two cuisines.

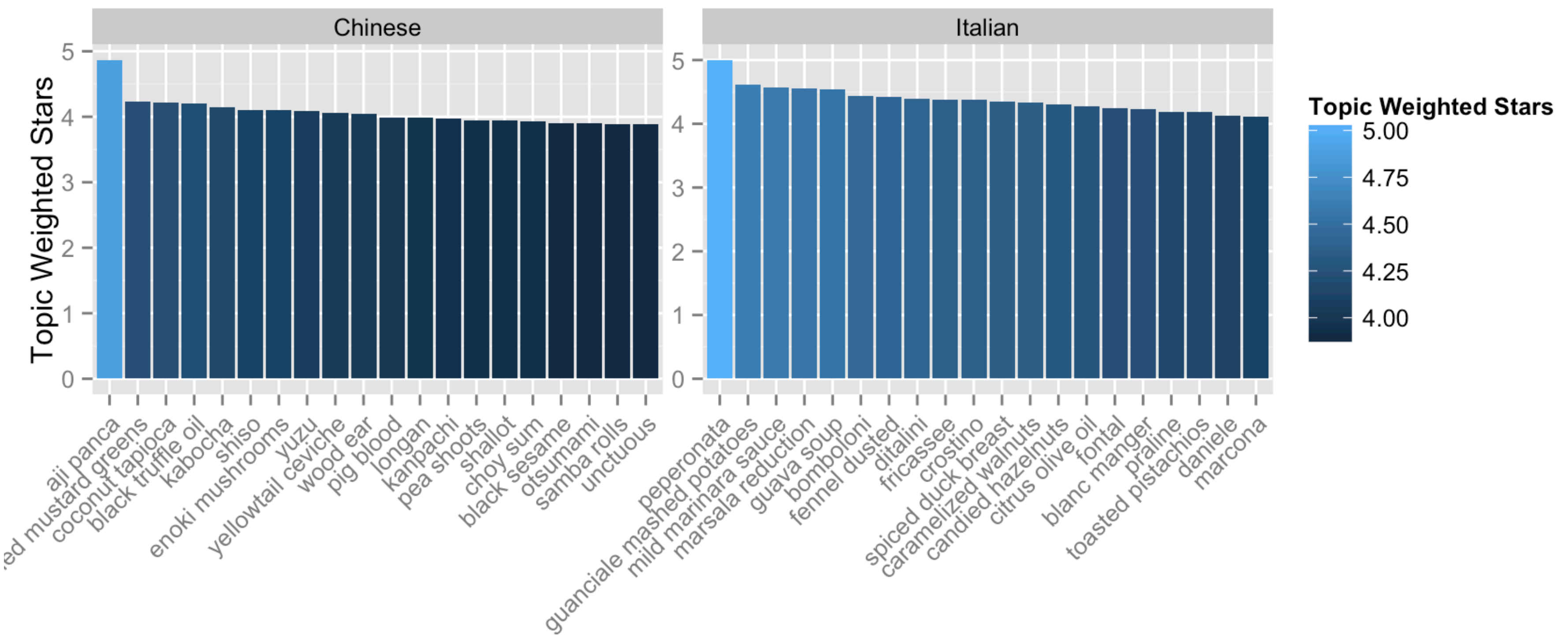

Figure 11. Dishes ranked by topic weighted average review stars for two cuisines.

3.3.3 Restaurant recommendation

To recommend a restaurant for a given dish, we need a score that tells us how good each restaurant is cooking that dish. We will use the same scores computed to rank dishes in the previous section, but we will aggregate them by restaurant to get a dishes-restaurants rating matrix.

The recommendation requested is in the form of a ranking listing the best N restaurants for a dish. We have followed two different approaches to computing our recommendations.

Given a dish, the first approach is simply to order the restaurants by decreasing order of their score for the dish. Then we present the top N restaurants.

A second approach contemplates the possibility of a dish being on a restaurant menu, but not being commented by some customer. We extend the list of restaurants recommended applying an user-based collaborative filter to predict for a restaurant the rating of a dish not commented in the restaurant reviews. In our collaborative filter, the dishes play the role of users, and the restaurants play the role of items. By applying a collaborative filter, we are assuming that restaurants that are good at cooking some dishes offer a similar quality in dishes that are related. In our work, the implementation of the collaborative filter is the one in the recommenderlab R package [9].

Results

We will show only restaurant recommendations using topic weighted average stars as the rating that one restaurant gets on a particular dish according to the customer reviews. In Figure 12 we can see the ranking of restaurants serving peperonata. As we can see, we have fewer than 20 elements because peperonata only appears in the reviews of 10 restaurants.

Could we infer how good would be the peperonata in other restaurants provided that they serve the dish? We can extend our list or recommended restaurants (Figure 13) using an user-based collaborative filter. As we can see, we expect that some restaurants that don't include a mention to peperonata in their reviews would cook the dish better than other restaurants that we know that they make the dish. The Yelp dataset does not include the menu of the restaurants, but we know that this information is also available on Yelp for some restaurants. Therefore in a real application we could further filter the ranking to keep only those restaurants that cook the dish.

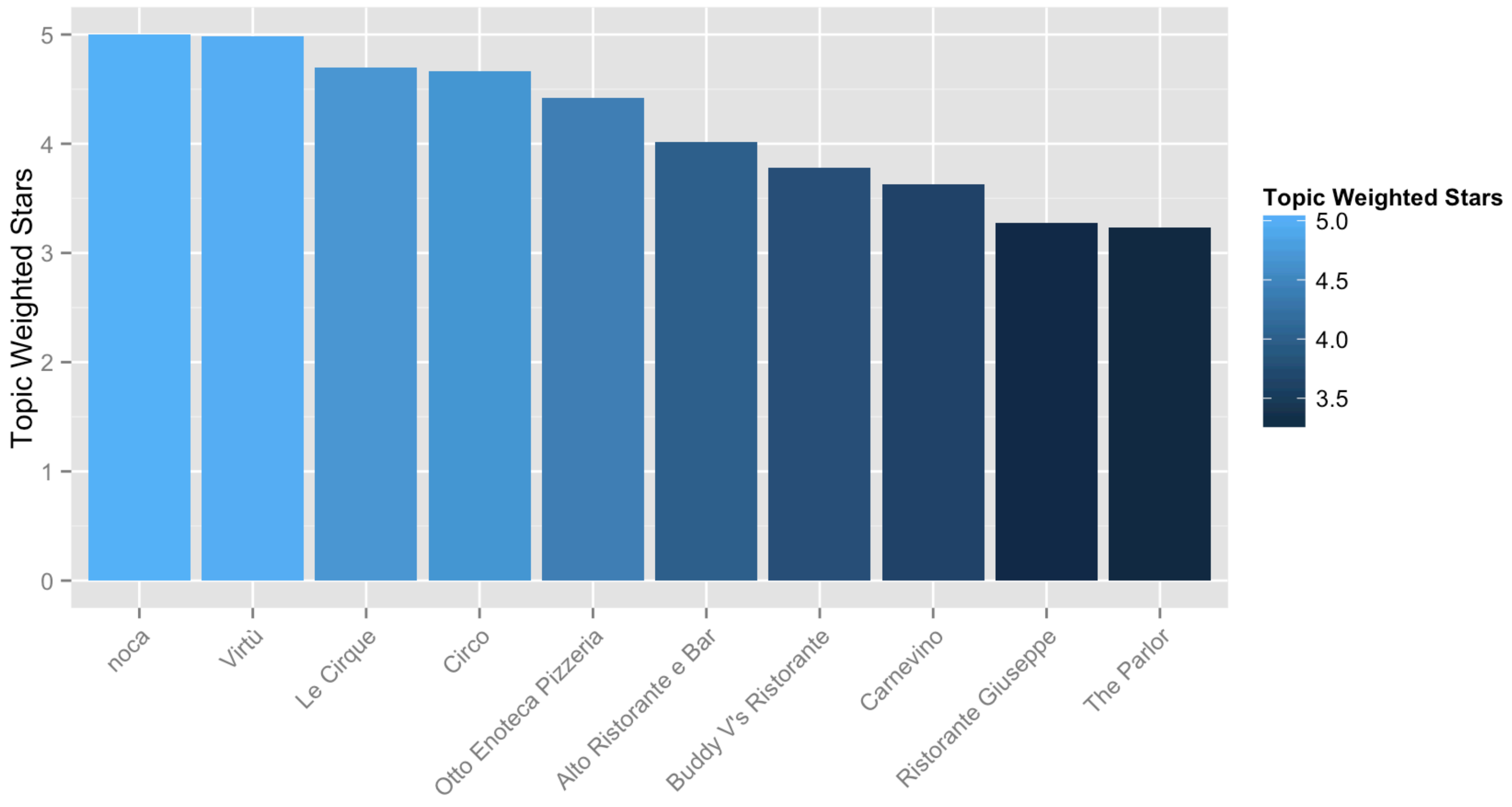

Figure 12. Restaurant recommendations for the Italian dish peperonata.

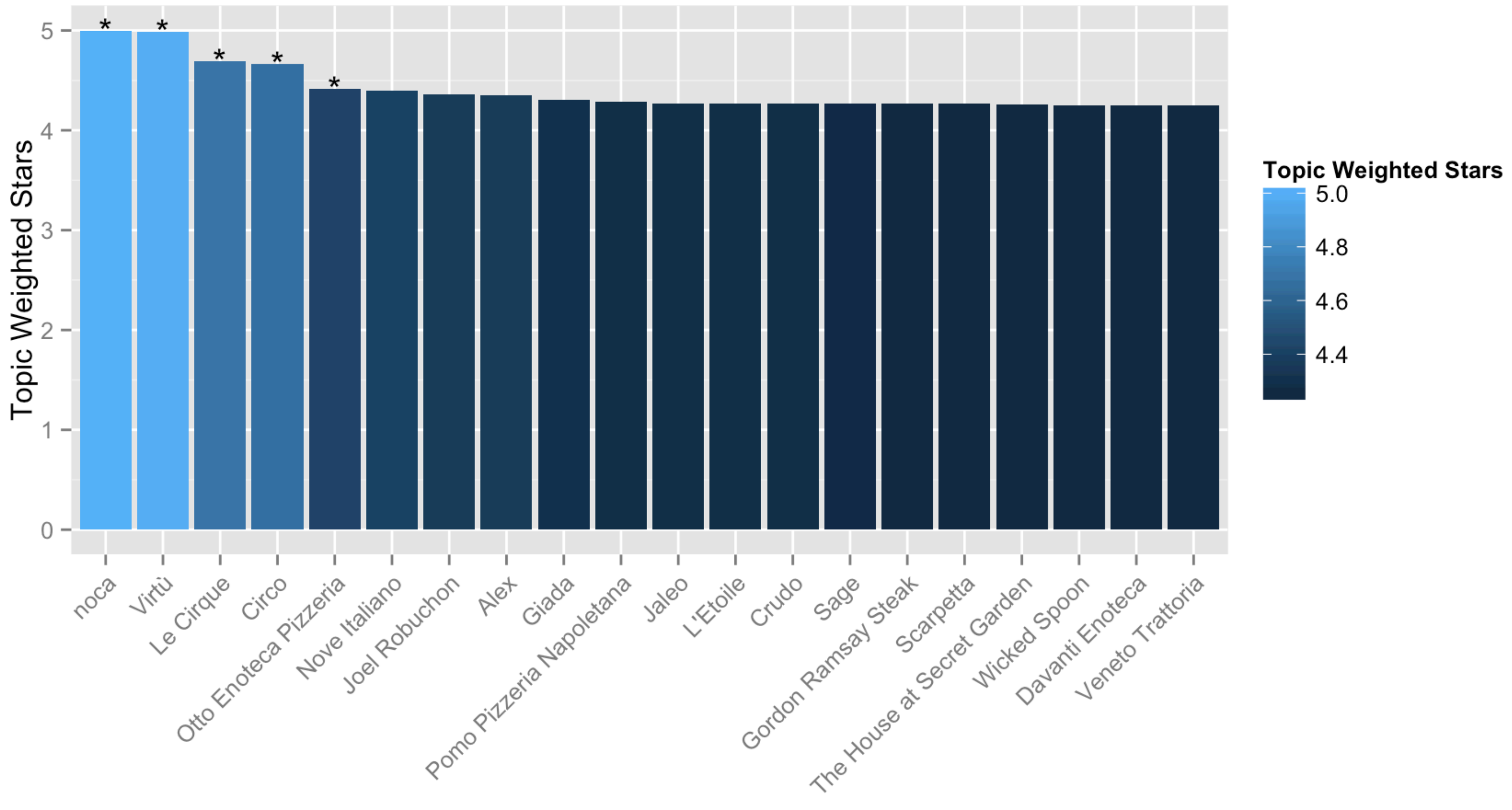

Figure 13. Restaurant recommendations for the Italian dish peperonata using aggregated topic weighted stars and an user-based colaborative filter. The stars on top of the bars indicate the restaurants that are known to serve peperonata.

3.4. Task 6. Restaurant hygiene prediction

3.4.1 The data

The data that we used in this task is slightly different. It is a corpus of 13,299 documents in which each document is the result of concatenating all the Yelp reviews of a restaurant. In addition to the reviews, we have the following additional data for each restaurant:

- cuisines offered.
- ZIP code.
- number of reviews.
- average rating in a 0 to 5 scale (5 being the best).

For 546 restaurants, we have a label indicating whether the restaurant has passed the latest public health inspection test (0) or not (1). We do not have a label for the remaining 12,753 (the evaluation set). Our objective is to use the 546 labeled records as the training set to train a classifier that predicts the label of the remaining 12,753 to predict whether they would pass the hygiene inspection or not. The performance of our classifier will be measured using the F1 measure.

One important feature of this dataset is that the training set has a balanced distribution of 0/1 labels, but the evaluation is unbalanced.

3.4.2 Classifier training

In training a classifier, two aspects are crucial: the supervised machine learning used to train our model and the features used as input.

Features extracted from reviews.

We preprocessed the reviews as follows:

- Remove non-writable characters.
- Strip extra white spaces.
- Lower case.
- Remove punctuation
- Remove numbers
- Stemming
- Stop words removal.

After that, each text was tokenized into unigrams, and the unigram frequencies were counted and stored into a document-term matrix of counts with a vocabulary of 13299 terms.

Term counts across all the corpus showed a typical Zipf distribution. We kept the most frequent terms that, summing all their frequencies, accounted for about 99% of the total number of words in the corpus. The resulting vocabulary has 13474 words. This reduced document-term matrix is the first set of text extracted features used for hygiene prediction.

To preprocess the text data and compute the document-term matrices we used the R packages “tm” [10] (v 0.6) and “RWeka” [11] (v 0.4-24).

To build the second set of features, we trained a topic model with 100 topics. We run a Latent Dirichlet Allocation algorithm (LDA) using the non-reduced document-term frequencies matrix as input. To estimate the model parameters we used a Gibbs sampling with a burn-in phase of 1000 iterations and later the distribution was sampled every 100 iterations during 2000 iterations. Then we used the matrix of topic probabilities for the documents as our text extracted features. We used the R package “topicmodels” [12] (v 0.2) to compute the topic models in this task.

A third approach consisted in training a 100 topics model using as input only the reviews in the training set that didn’t pass the hygiene inspection. The idea behind this approach is to extract specific features for the minority class to increase our specificity. Once we had the model, we computed the posterior probabilities over the remaining reviews in the training and test set, and we built a matrix of document-topic probabilities similar to the one mentioned above.

Additional features

Together with the reviews, other features were provided: average rating, the number of reviews, cuisine categories and ZIP code. We used average rating and number of reviews as regular numeric features. The ZIP code was encoded as a factor and finally the cuisine categories were encoded using a set of dummy variables that took a value of 1 for the categories that the restaurant belonged and 0 for the others.

Training algorithms

This task is a supervised machine learning classification task. We have selected three algorithms to train our models: random forest, SVM with polynomial kernel and logistic regression.

In all cases, we built a matrix of features concatenating each of the text features with all the additional features. We hold 20% of the training cases to evaluate the performance of our approaches (test set), and we used the remaining 80% to train our classifiers.

To train a classifier with any of the algorithms mentioned above, we used 10-fold cross-validation using the training set for model parameter selection. Once we selected the model parameters, we trained our final model over the complete training set.

We have used the R package `caret` (v 6.0) [13] to train our classifiers. Concretely, the method parameter passed to the function `train` that we used were:

- `rf` for random forest.
- `svmPoly` for the SVM with a polynomial kernel.
- `glmStepAIC` for logistic regression.

We have held out a 20% of the training data as a test set in which we can perform our evaluation of the algorithm performance.

We have assessed the performance in two ways: graphing the receiver operating characteristic (ROC) curve for our prediction over the test set and computing the F1 measure for the prediction on the test set. To compute the measures of performance we used the R package `ROCR` [14]

Results

Figure 14. shows the ROC curve for Random Forest using five combinations of features:

1. Reduced document-term matrix + additional features (unigram)
2. Topic model with 50 topics + additional features (TM50)
3. Topic model with 100 features for label 1 cases + additional features (TM_label1)
4. Topic model with 50 topics + topic model with 100 features for label 1 cases + additional features. (TM50+TM_label1)
5. Reduced document-term matrix + Topic model with 50 topics + topic model with 100 features for label 1 cases + additional features. (unigram+TM50+TM_label1)

Figure 15 shows the same information for SVM. Figure 16 show the ROC curve for logistic regression, but only for combinations 2 and 3.

Figure 17 shows the corresponding F1 for each algorithm and features combination.

As we can see, random forest using a combination of all the features obtained gives the best result. Nevertheless, although the F1 measure on the hold-out test set is close to 0.8, the F1 score obtained in the evaluation set on the Coursera platform was only 0.5577.

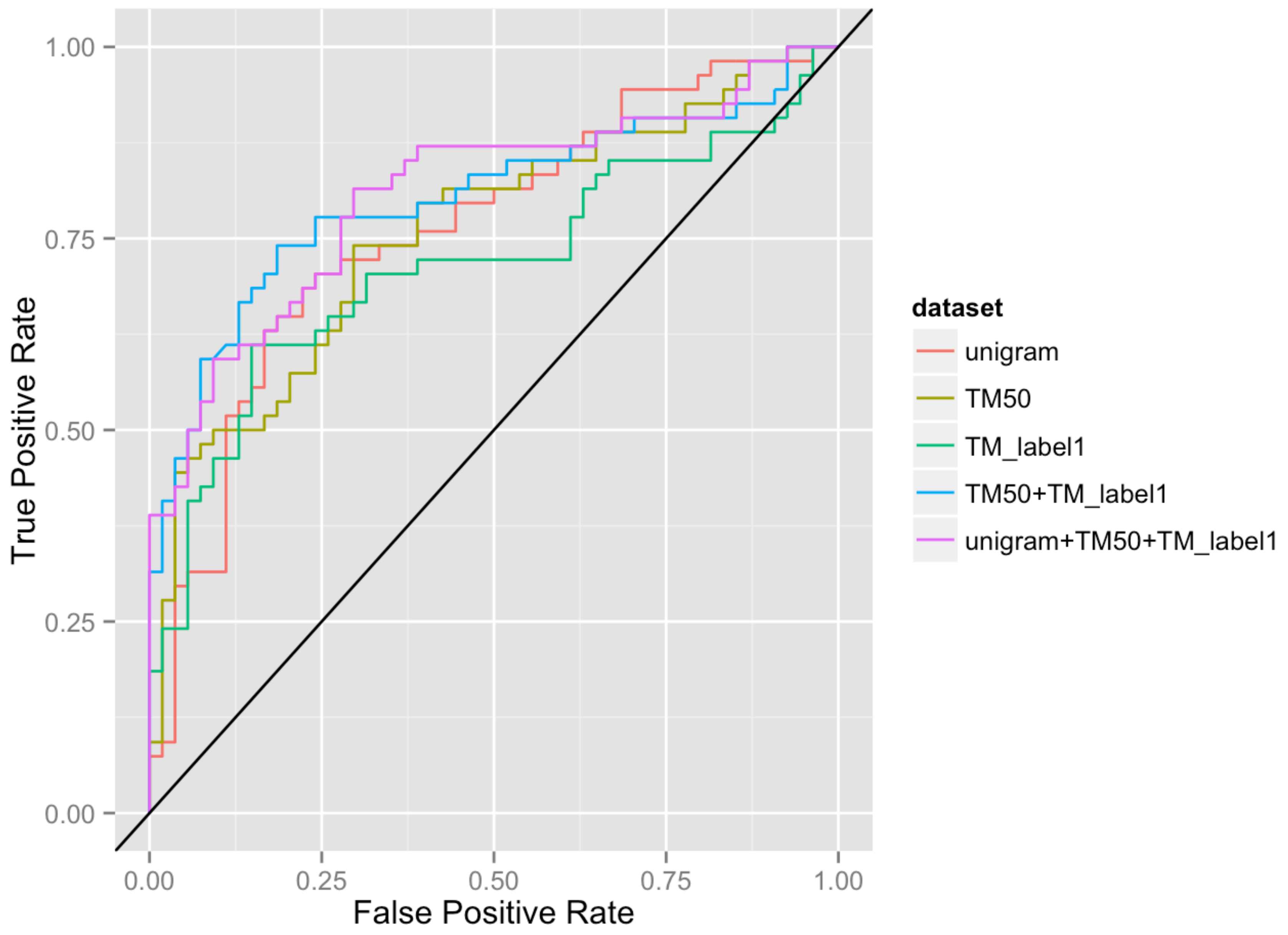

Figure 14. ROC curve for classifiers trained using random forest and different features combinations.

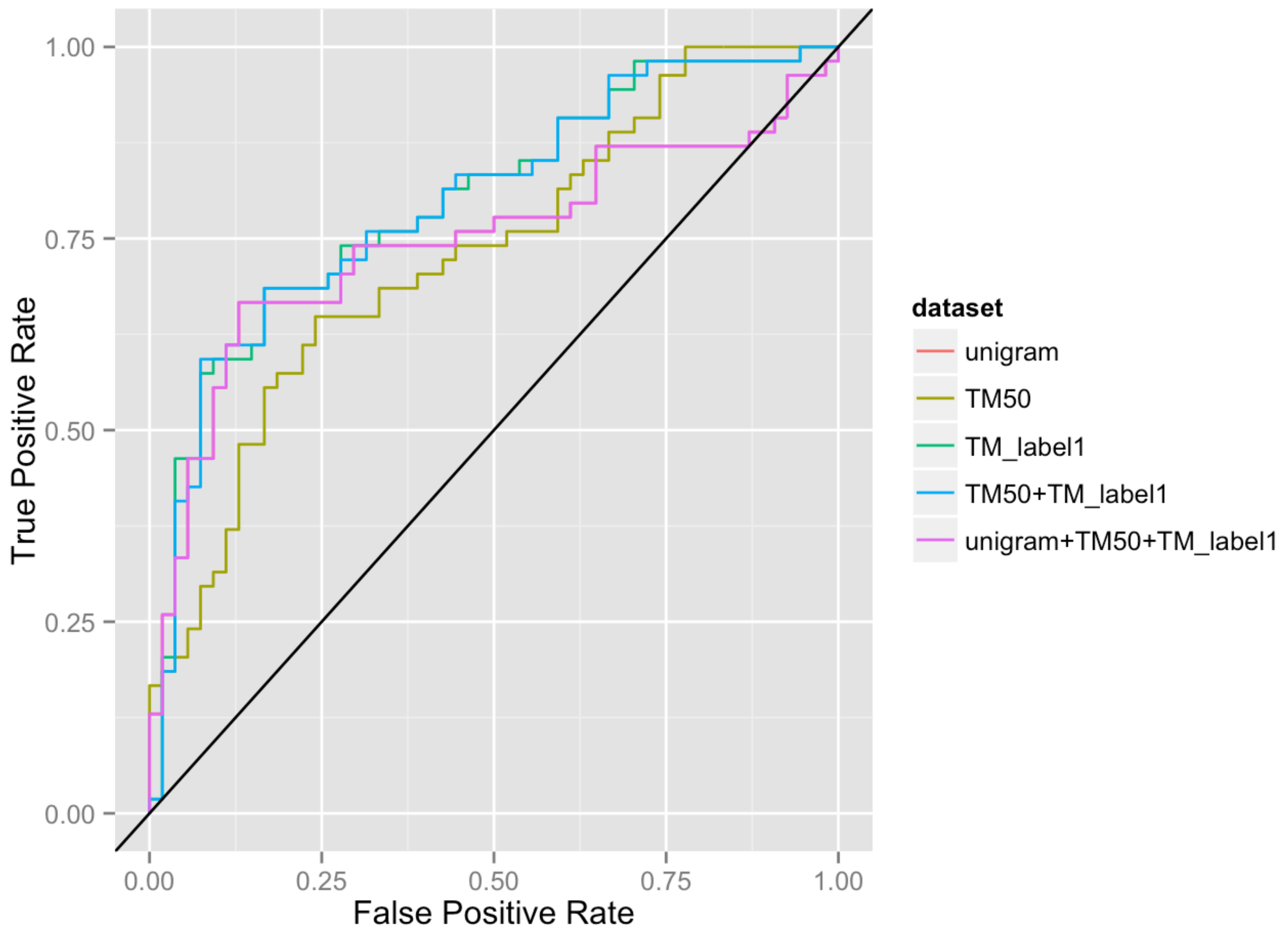

Figure 15. ROC curve for classifiers trained using SVM with polynomial kernel and different features combinations.

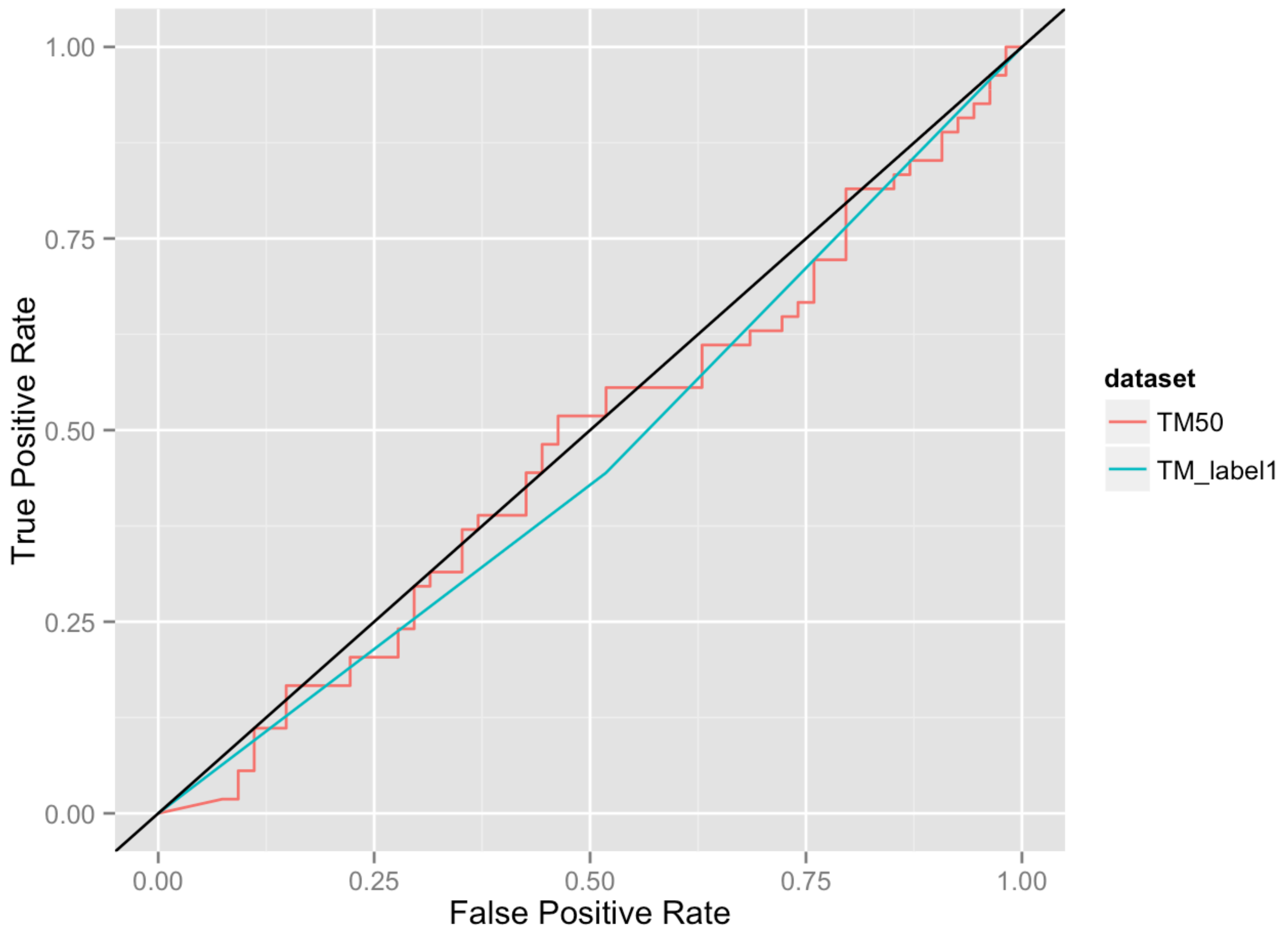

Figure 16. ROC curve for classifiers trained using logistic regression and different features combinations.

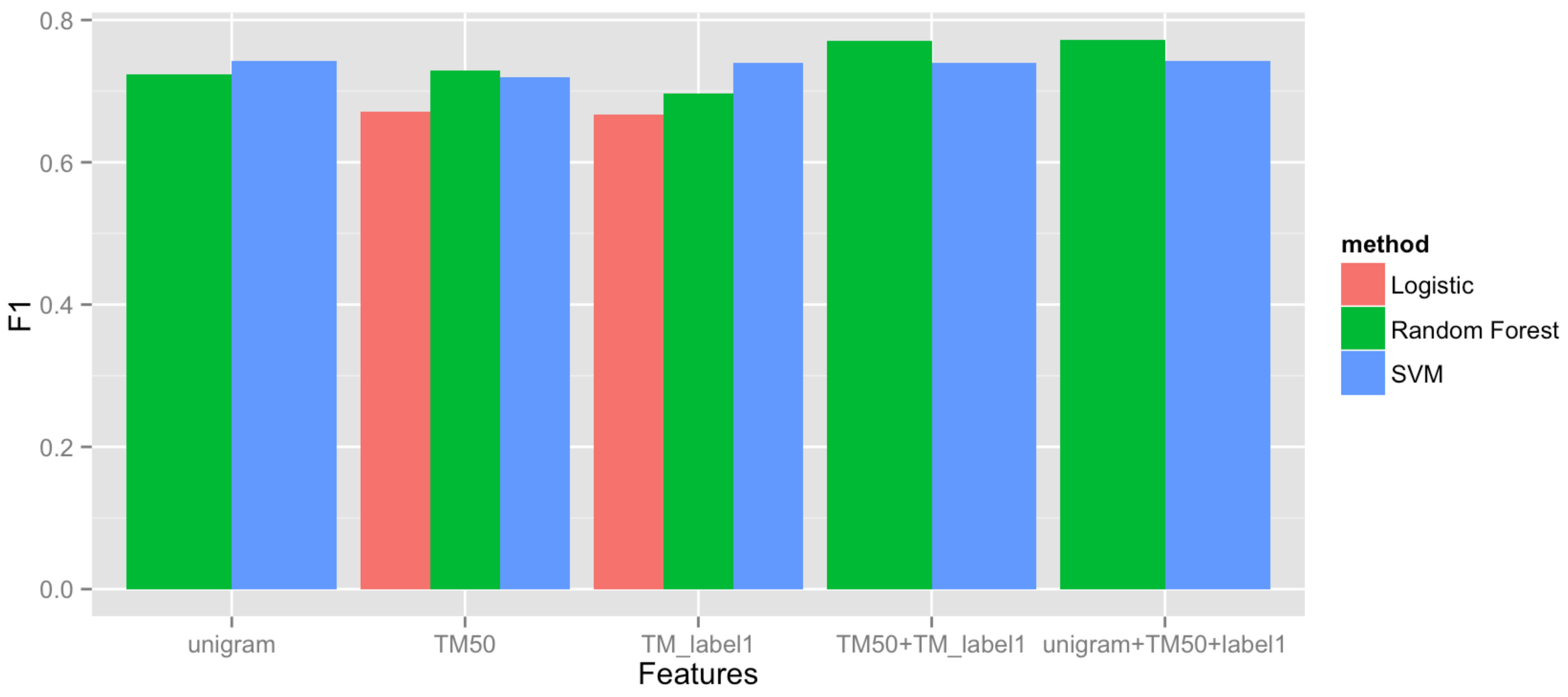

Figure 17. F1 score computed on the hold-out test set for all the classifiers trained.

A closer exam of the prediction errors on the hold-out test set shows that 91% of the mistakes were caused by false negatives. An exam of the reviews does not show any sign of complaints about the restaurants hygiene. Therefore, the dataset does not provide enough information to predict those cases.

4. Usefulness of the results.

4.1 Topic models

Topics frequently treated in Yelp reviews reflect what people care when they go to a restaurant. This information is important for restaurant owners because it helps them to focus their efforts to improve their business in the aspects of their customers experience that matter. In particular, the exam of the topics in the model obtained from negative reviews point to what a restaurant owner and staff shoud avoid at all costs like long waiting times.

4.2 Cuisine map

There are many cuisines, and customers often only know a few of them and usually do not try new cuisines because they do not know what to expect and some reticence of moving outside their confort zone. Our map of cuisines can help users to explore new cuisines by starting at their favorite cuisines and then trying neighbor cuisines that could discover new culinary experiences that are familiar enough to keep them in their confort zone. On the other hand, people who seek a radically different experience can jump from cluster to cluster in our map.

4.3 Dish ranking and restaurant recommendation for a dish

We have built a system that ranks dishes within a cuisine by popularity and that provides a recommendation about where to try those dishes. These products are useful for users who are looking for new culinary experiences and for restaurant owners who want to modify their menu offer. However, these products could be useful also for Yelp. Restaurants often list their menus in Yelp. We can use the dishes listed in the menus to go beyond the explicit information provided in the reviews and recommend restaurants for a dish, even if the dish was never mentioned in the restaurant comments.

4.4 Hygiene prediction

The capacity of predicting whether a restaurant will pass a hygiene inspection from the contents of user reviews is a valuable tool for governments. By monitoring the opinions of customers, a flag can be raised when the content of the reviews indicates that the restaurant may be deviating from public health regulations. This in turn helps to focus public resources and makes public inspections more efficient.

5. Novelty of Exploration

We have applied a set of tools for text mining along this project. In all the cases, we have tested different sets of parameters and data-algorithm combinations to seek for optimal results.

Also, during the development of this project some standard tools have been provided. For the most of this project, we have not used the tools provided but looked for equivalent solutions in the R environment.

In addition to that, we have applied some approaches not suggested in the tasks descriptions. The next sections cover these approaches.

5.1 Topic modeling as a tool for dimensionality reduction and feature creation.

Although we have used topic modeling in a straightforward way in task 1, later we have used LDA topic modeling as a tool for dimensionality reduction and feature creation.

A bag-of-words representation of a corpus is a rich source of information about the documents in the dataset. Nevertheless, the number of terms in a bag-of-words can grow until dimensions that prevent efficient computation using a document term matrix. On the other hand, terms can have multiple meanings depending on the context. Topic modeling is a tool that helps to reduce both problems by providing a conceptual representation of the documents in a reduced space.

Among other parameters, LDA topic modeling computes the posterior probability of each document covering each of the k topics in the model. Usually k is much smaller than the number of terms in the corpus vocabulary. Thus, the probabilities are a vector representation of the documents that summarizes the concepts covered in the document in a much smaller dimension space.

Also, we are no longer considering terms as individual entities, but as part of a concept. Each term in a concept can provide a contribution to several topics, being the most probably meaning of the term that related to the most probable topic of the document.

Although this idea is not new [15], the application of topic modeling in this way is novel enough not to be the first option for everyone.

5.2 Extract a sparse graph representation from document similarities.

Representing a graph as a similarity matrix, then applying some transformation to the matrix and returning to the graph representation is a common approach in network analysis. However, what is not so usual is to start from a dense similarity or correlation matrix that is not related to a graph in origin and, then, obtain a sparse graph representation that provides a map for relevant relationships in the similarity matrix.

Let’s say that we have a similarity matrix computed by computing the cosine similarity of the vector representation of the documents in a corpus. The novelty of our approach is to think of that matrix as the representation of a dense graph in which the documents are the nodes, and the edge weights are the similarities between document.

To discover the most important relationships among documents, the backbone network, we need to remove edges systematically until we keep only significant similarities. For that we have applied the method described in [7].

6. Contribution of New Knowledge

6.1 Using topic similarity to discover relationships among groups.

Sometimes we can have a corpus with documents being originated or associated to different categories of items. For example, in the present project we have reviews associated with restaurants that in turn are classified in different cuisine categories. Other contexts are emails sent by different staff categories, articles in different newsgroups or posts labeled with different tags.

Those groups can be represented as the nodes in a graph and their relationships as edges. The edges can represent a direct interaction between nodes (person in group A sends an email to person in group B, paper in discipline A refers to paper in discipline B) or sharing some explicit feature (tags in common from posts in group A and B, papers in different disciplines with authors in common). These explicit representations of the relationships among groups can be studied using well-known measures in social network analysis like betweeness.

Latent representations of the documents in a corpus also provide a means to study the relationships among groups in a graph. One of those non-explicit representations is latent topics covered by the documents in the corpus. We can think that two groups are similar if their documents asociated cover similar topics. The similarity in the topics covered in the documents asociated to two groups can be used as a measure of the strengh of the relatioships between the groups. Then we can select the strongest relationships to build a graph in which the groups are connected only by relevant relationships. This kind of graph generated by a latent space model allows us to discover hidden asociations among the groups represented by the topics that they cover in their communications.

The work done on this project for Task 2 applyies this idea of using document topics as the means to discover the backbone graph of the relationships among cuisine types (Figure 6). We have projected each cuisine type into a laten space in which the dimensions are the topics of the documents asociated with each cuisine. The strength of the relationship between any two cuisines is then determined by how close they are in this new space. Finally, we select the most relevant links between cuisines based on their statistical significance.

Our model is a simpler version of a Relational Topic Model [16]. We have replaced the logistic regression approach to predict the links among documents by the statistical significance of the cosine similarity among the documents.

6.2 Using a collaborative filter to infer the skill of a restaurant at cooking a dish.

Recommender systems implement user-based collaborative filters to infer the rating that a user will give to a product. Collaborative filters based their prediction on the preferences of the user on other items and in the preferences of other users with a similar profile.

In task 5 we used a user-based collaborative filter for a restaurant recommendation. We replaced users with dishes and items with restaurants. In this context, we can interpret a rating as how skillful is one restaurant at cooking the dish. We can say that two dishes require a similar skill profile if the same restaurants cook them with the same degree of success in the same restaurants. For example, let’s say that restaurants 1 and 2 serve a good dish A, but restaurants 4 and 5 are not good at cooking the dish. On the other hand, let’s say that dish B is well cooked in restaurants 1, 2, and 3 but not in restaurants 4 and 5. We can assume that the set of skills needed to cook both dishes is similar and infer that dish A will have a good quality in restaurant 3.

This approach allows to extend a list of recommendations by including restaurants that allegedly should serve a good dish, but there is not mention to the dish in her reviews. Nevertheless, the list of recommended restaurants requires further filtering only to include those that actually serve the dish. We can do this using an external source of information like the menus that the restaurants often list in Yelp.

7. References

1. M. Anderson and J. Magruder. “Learning from the Crowd.” The Economic Journal. 5 October,2011. (http://are.berkeley.edu/~mlanderson/pdf/Anderson%20and%20Magruder.pdf)

2. Blei DM, Ng AY, Jordan MI (2003b). “Latent Dirichlet Allocation.” Journal of Machine Learning Research, 3, 993–1022. (http://robotics.stanford.edu/~ang/papers/jair03-lda.pdf)

3. Griffiths TL, Steyvers M (2004). “Finding Scientific Topics.” Proceedings of the National Academy of Sciences of the United States of America, 101, 5228–5235. (http://dx.doi.org/10.1073/pnas.0307752101)

4. Phan XH, Nguyen LM, Horiguchi S (2008). “Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-Scale Data Collections.” In Proceedings of the 17th International World Wide Web Conference (WWW 2008), pp. 91–100. Beijing, China. (http://dx.doi.org/10.1145/1367497.1367510)

5. Blei DM, Lafferty JD (2007). “A Correlated Topic Model of Science.” The Annals of Applied Statistics, 1(1), 17–35. (http://dx.doi.org/10.1214/07-AOAS114)

6. Lv, Y., Zhai, C., 2011. Lower-bounding term frequency normalization, in:. Presented at the Proceedings of the 20th ACM international conference on Information and knowledge management, ACM, pp. 7–16. (http://dx.doi.org/10.1145/2063576.2063584)

7. Serrano, M.A., Boguna, M., Vespignani, A., 2009. Extracting the multiscale backbone of complex weighted networks. arXiv. (http://dx.doi.org/10.1073/pnas.0808904106)doi:10.1073/pnas.0808904106 (doi:10.1073/pnas.0808904106)

8. Raghavan, U.N., Albert, R., Kumara, S., 2007. Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E 76, 036106. (http://dx.doi.org/10.1103/PhysRevE.76.036106)

9. Recommenderlab in CRAN (https://cran.r-project.org/web/packages/recommenderlab/index.html)

10. Ingo Feinerer, Kurt Hornik, and David Meyer (2008). Text Mining Infrastructure in R. Journal of Statistical Software 25(5): 1-54 (http://dx.doi.org/10.18637/jss.v025.i05)

11. Kurt Hornik, Christian Buchta, Achim Zeileis (2009) Open-Source Machine Learning: R Meets Weka. Computational Statistics, 24(2), 225-232. (http://dx.doi.org/10.1007/s00180-008-0119-7)doi:10.1007/s00180-008-0119-7 (doi:10.1007/s00180-008-0119-7)

12. Hornik, K., Grün, B., 2011. topicmodels: An R package for fitting topic models. Journal of Statistical Software 40, 1–30. (http://dx.doi.org/10.18637/jss.v040.i13)

13. Caret package (http://topepo.github.io/caret/index.html)

14. ROCR package at CRAN (https://cran.r-project.org/web/packages/ROCR/index.html)

15. Crain, S.P., Zhou, K., Yang, S.-H., Zha, H., 2012. Dimensionality Reduction and Topic Modeling: From Latent Semantic Indexing to Latent Dirichlet Allocation and Beyond, in: Aggarwal, C.C., Zhai, C. (Eds.), Mining Text Data. Springer US, Boston, MA, pp. 129–161. (http://dx.doi.org/10.1007/978-1-4614-3223-4_5)doi:10.1007/978-1-4614-3223-4_5 (doi:10.1007/978-1-4614-3223-4_5)

16. Chang, J., Blei, D.M., 2009. Relational topic models for document networks, in:. Presented at the International Conference on Artificial Intelligence and Statistics, pp. 81–88. (http://jmlr.org/proceedings/papers/v5/chang09a.html)