Data Mining Project

by University of Illinois at Urbana-Champaign

Restaurant Analysis – YELP Reviews

Student: Begoña López Rodríguez

Date: December 10, 2019

INDEX

1 Executive summary

In this final course project of the Illinois University Data Mining specialization, I have been able to put into practice all the acquired knowledge during the last 6 months, through the different subjects or specialized taught fields. I would like to emphasize that this capstone was not only an exercise to consolidate concepts and acquire practical skills on specific tools, but has also allowed me to globally understand the kind of problems I may face with data mining, how powerful is text mining, and how it allows us understanding any scenario or context, where digital conversations take place, and how we can build systems to analyze all types of digital context and extract relevant information that helps people in their daily decisions.

The project targets to analyze a restaurant recommendations data set to extract relevant information, used to build **specific functionalities that could be included in a supposed recommender system,** and would help users to **make decisions about choosing a kind of cuisine, dishes, restaurants, even building a predictive system to anticipate whether a specific restaurant would pass or not a health inspection.** For this, they provide us with a real data set from the YELP social network that contains 1,125,458.00 global reviews plus all the information this network collects from its restaurants and users.

In this report, I detail all the steps I have followed for the data mining, specially focusing on the comments or reviews mining, modeling or building the **new recommendation functionalities** specific engines. The main addressed challenges I have faced in each section, the developed strategies and the achieved results are summarized as follows.

Point out, in concerns to a more technical aspect, that I have perform the entire project with R, with the purpose of exploring the different packages implemented state of art in this language state.

1 Exploration of the data set

I start the project by doing a data set generic exploration to understand the conversations context, facing a further development of the different specific functionalities. I structure this previous analysis in three steps, each one with a higher detail level:

1.- I use the **LDA topic modeling** to identify the **main conversations existing in the data set**. This first approach without filter. This means, I apply LDA to the total 1,125,458 reviews set.

In order to analyze the results in a simpler way, I create two visualizations: one more global, hierarchical and with a circular layout, which gives me a global view of the most frequent terms for each topic; and another containing more detailed information, in which I used the 100 terms per topic word cloud format.

Regarding the obtained results, highlighting how the identified topics clearly show conversations that talk about specific types of cuisine, being in the same topic those of more similar kind to each other.

2.- In a second approach, I try to identify which are the existing **different conversations** in the reviews, that rate the **experience** as **positive, versus** the ones that rate as **negative**.

In this case, I do a previous analysis to identify how to classify in a more adequate way the positive and negative reviews, as well as which is the most relevant subset for the analysis, depending on the existing level that exists in positive vs negative.

Once the subset has been chosen, I apply the same techniques from the previous point, LDA and the two defined visualizations types to analyze the results.

As relevant conclusions I draw that there are recurring words for all topics, both positive and negative (service, time, quality, price…), concluding that these are always rated concepts when a restaurant experience is evaluated.

3.- Finally, in order to **give context** to the most frequent terms and/or those I have found more relevant, I perform a frequency **analysis by tri-gram** structure type (three adjacent words).

The obtained results clearly show the differences between positive and negative rated conversations.

2 Cuisine clustering and map construction

The goal of this section goal is building a **similarities map among the different types of cuisine that allows a user to find new kinds of cuisine,** unknown for him but similar to others he knows and likes.

I decide to approach the exercise from **two different strategies** which, in my opinion are complementary.

- On the one hand, searching for **conversations or reviews similarities**, as set out in the exercise.
- On the other hand, although it was not requested in the exercise, searching for **similarities according to the users**; this is, according to the visit frequency (or published review) to restaurants of different cuisine type.

After choosing a limited cuisine types subset (selecting the most relevant), I prepare a Heat Map to view the similarities applying various algorithms for the distance calculation: Jaccard and Cosine. After, I apply TF-IDF (in order to penalize the most frequent words in the cuisine type itself, and also the most frequent in different cuisine types), and the LDA algorithm, which models topics. Finally, I apply clustering techniques, using the K-Means algorithm, both with Jaccard and LDA. In this report, I include all the generated Heat Maps and the performed analysis showing the obtained results. Highlight how the different explored techniques have been generating clearer and more precise results.

Secondly, I have made the similarities Heat Maps according to the **users preferences**; that is, frequencies of visits to restaurants of certain cuisine types. In this case I also apply the K-Means clustering.

Comparing the obtained results in the two analysis ways, it is clearly verified how the first classification revolves around specific cuisine types characteristics (ingredients, flavors,…), while the second focuses on users preferences, with more specific segmentation of a more youthful, informal cuisine type, etc.

3 Dish recognition

Once identified the cuisine types similarities to the ones we like , we move on to identify **which dishes are typical for that kind of cuisine.** The purpose of this section is to build a **dish recognizer.** That means an engine that, given a specific cuisine type, undermines the conversations or reviews to identify typical dishes.

I use **the R** wor2vec function to build this engine, that uses Mutual information techniques in order to identify in a given corpus, similar words to the given one or ones in a list. This function builds a trained model based on a given word list. The more extensive and specific this list is, the better the model will be. Once modeled and trained, it report similar words to a specified word or word list.

To perform the exercise, which by the way, I decide to pick the Italian cuisine, I start from the list given by the course with the initial Italian cuisine dishes. Firstly, I manually debug the list (containing false positives and negatives), and I repeatedly apply modeling and training, enriching the base list in each interaction. Once the two interactions have been done, the obtained results are surprisingly good, so I do not feel it is necessary to continue building a more robust model. I would like to highlight the achieved precision level. Testing it with the word “dessert**” I specifically obtain Italian desserts with a 100% effectiveness (in the top 20 with higher MI values).**

Remark that, in my opinion, this algorithm really works well when the context we are looking at is very limited. Its operation is based on the fact that two words are closer similar the more similar the words surroundings those are. In the case in question is ideal to apply this, since the conversations are specific of restaurant experiences and the comments to specific dishes usually are in very similar contexts.

4 Popular dishes and restaurant recommendation

Next step is to build a **popular dish ranking for a specific cuisine type,** as well as a **restaurant recommender to try a specific dish.**

For both functionalities or ranking, popular dishes and restaurant recommendations, I must identify the variables of our set data which influences on those rankings (making a dish more popular of a restaurant more recommended for a specific dish), and modeling how those variables behavior must be in the rankings.

Regarding the ***popular dishes ranking*** I identify three concepts I understand we must take in consideration:

- **Popularity**. How much is said about that dish?
- **Perception**. If they talk well or bad about it?
- **Specialization**. Is a usual dish or only serve in very specific places?

In the current report I reason which variables I understand qualify these aspects, and how I understand they should behave in the ranking. The resulting model is:

Ranking(d) = **α** * *Sum(Occurrences(d)) / TotalRest* + **β** * *Mean (Sentiment(rew) * (1 + log (1 + #VotesUseful(rew)))* + **μ** * *log ((Sum(Occurrence(d)) + 1) / DistinctRest(d))*

Popularity(p) | Perception(p) | specialization(p)

Emphasize that in regard to perception or feeling, I have deepened the use of R Sentimentr, that allows analyze the feelings by sentences or phrases. I consider that standing with the user overall rating or qualification on the experience reflects the overall experience assessment, which may differ from the experienced and expressed specific perception on the specific dish.

With regard to **r*estaurant recommendation ranking*** the modeling I have done is as follows:

Ranking(d, rest) = **α** * *Sum(Occurrence(d, rest)) / Sum(Occurrence (d))* + **β** * *Mean (Sentiment(rew, d. rest) * (1 + log (1 + #VotesUseful(rew, d, rest)))*

Specialization(p, rest) | Perception (p, rest.)

For both ranking, I have done visualizations that show in a very visual way both the ranking results, as well as the different aspects considered in the models (popularity, perception and specialization).

Regarding each of the aspects' weight that make up the models, I have carried out several tests with α, β and μ; and α and β respectively parameters until fine tuning the weights adjustment.

5 Predictive model. Public health inspections

As a final step, we model a predictive system to anticipate where a particular restaurant would pass or not a health inspection. Hygiene and health issuer are a key aspect for most people when deciding either to eat or not in a restaurant. Having a functionality that predict this, based on the restaurants data (conversations or reviews, restaurant ratings, etc.), if this is going to have a minimum quality level is essential.

To get this predictive model we have the 13.299 restaurants reviews plus an additional data series (postcode, medium rating, number of reviews and cuisine type). Also, we have the tags indicating if these 546 restaurants have pass or not the health inspection. This sub set allows me training the different predictive algorithms I explore, and finish building the predictive model.

The part I found more interesting about this exercise was the first part, the Engineering Feature. In this part, in which we chose and prepare the data that will be part of our model, I explored different strategies to select the model features or characteristics, which I believe led me to improve its accuracy. These strategies have been:

- Unigram and bigram analysis, applying TF-IDF. Here I have used filters by dispersion levels (or lack of data) above 90 and 95% with the purpose of obtaining the greater value model variables.
- Applying Mutual Information (word2vec), as we did in the *dish recognizer* section, to identify which exact terms have to do with related issues to hygiene and cleanliness.

I have applied the following predictive algorithms in all these cases:

- **Logistic Regression**
- **Decision Tree**
- **Random Forest**
- **Xgboost**

In order to check the accuracy of the different predictive models that I have been modeling, and since I had no labels or method of checking the accuracy for the rest of the restaurants, I have considered 80% of the 546 restaurants with known results (label) for the training of the model, and the rest of 20% for the predition and measurement of precision. Specifically, I calculated parameter F1. The result is the one I show in the following table:

	Logistic Regression	*Decision Tree*	*Random Forest*	*Xgboost*
Words. TF-IDF	0,5454	0,4597	0,6725	0,5192
Bigram	0.6955	0,5333	0,6153	0,5591
Mutual Information	0,5625	0,5057	0,6061	0,5072

We have obtained the highest values for the Logistic Regression algorithm, followed by the Random Forest algorithm.

It makes sense, since they are algorithms that work best when most of the features we analyze are linear and not categorical, as is the case we are analyzing.

On the other hand, I was a little surprised that, as regards the strategy of Feature Engineering of Mutual Information (words related to cleaning issues), it has generally given less accurate results. I interpret this as that the issues of cleanliness and hygiene do not usually, except in very extreme cases, be a precise object of conversation, and we must look at other parameters of the conversations such as poor quality, etc.

2 Exploration of the data set

2.1. Extracting topics

To start with, as a preliminary exercise, I explore the dataset and decide to carry out a first analysis of the whole dataset, without previous filters, in order to understand what topics or conversations are widely discussed on this social network. I chose the tool R, and apply the **topic model LDA** with the following particularities:

- I apply **stop words** to remove irrelevant common words for this analysis.

- I search **10 topics.** Considering it's a sufficiently representative number to understand the discussions that are currently taking place.

- I use the graph () function to create a graph showing the **15 most repeated words of the respective topics**, distinguishing each topic by a color, and using the opacity of the node to indicate the weight of that word in the topic. The chart type is hierarchical and the layout circular.

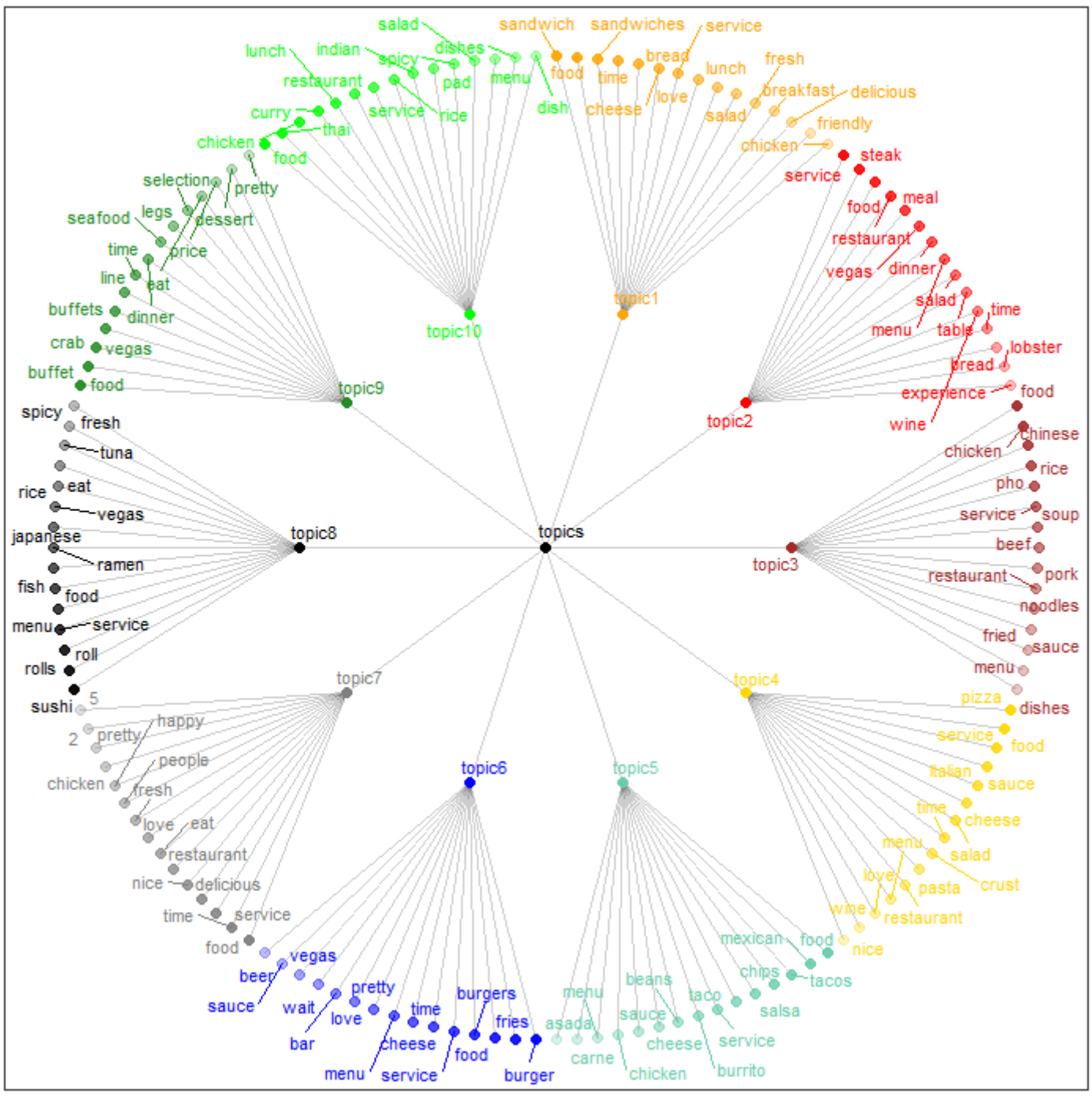


Figure 1. Topics YELP. LDA topic model

We clearly identify that the different topics refer to specific types of cuisine: topic1 (sandwich), topic2 (steak), topic3 (Chinese), topic4 (pizza/Italian), topic5(Mexican), topic6(burger), topic7(general), topic8(Asian), topic9(buffet), topic10(Indian). We observe that all topics, except topic9, have the word ***service*** among their 15 most used words; as well as the word ***time***, among the 15 most frequent for all topics excluding topic3, topic5, topic8 and topic10. We understand that both concepts, *service* and *time*, are recurring when we evaluate an experience.

In order to make a deeper analysis of these topics, we generate **word clouds** with the 100 most used for each topic

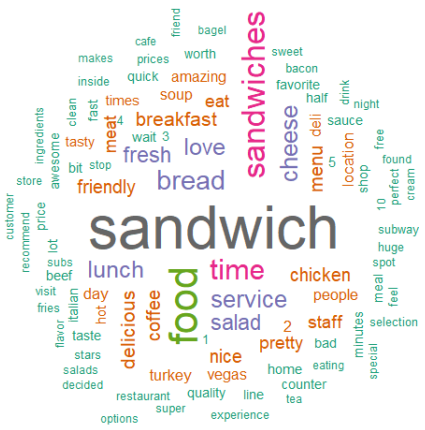


Figure 2. Topic1

Figure 3. Topic2

Figure 4. Topic3

Figure 5. Topic4

Figure 6. Topic5

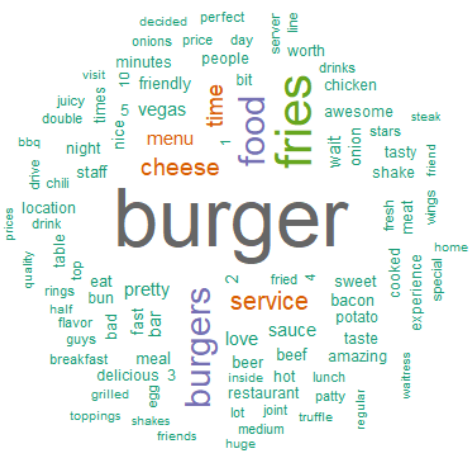


Figure 7. Topic6

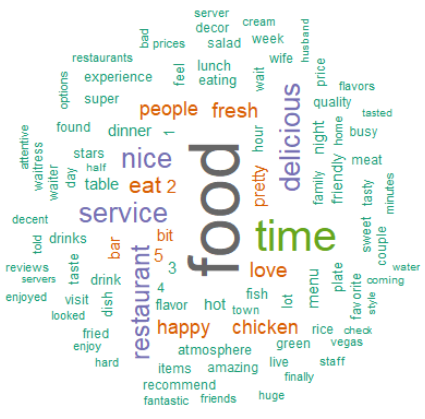


Figure 8. Topic7

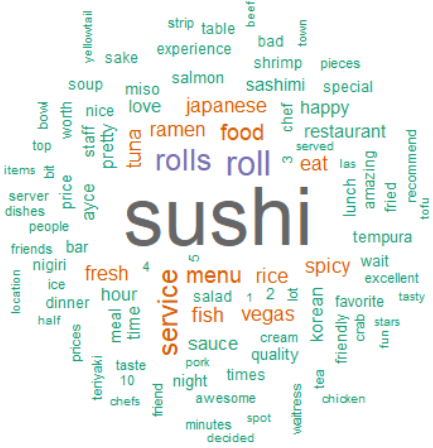


Figure 9. Topic8

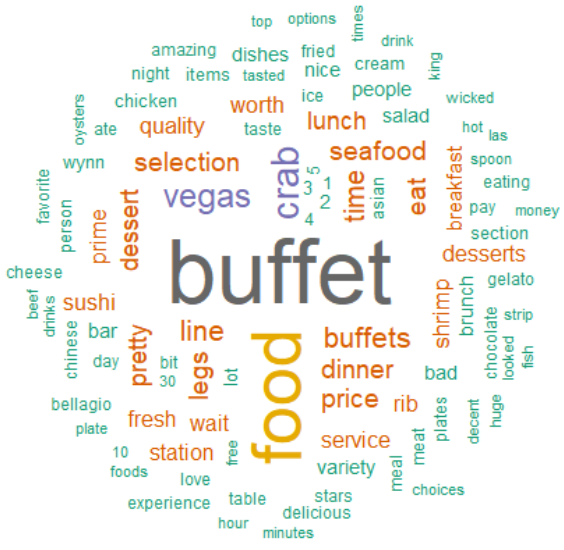


Figure 10. Topic9

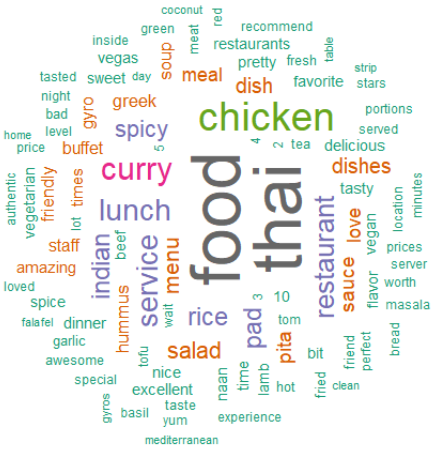


Figure 11. Topic10

We draw some very interesting conclusions from this analysis:

- We reaffirm the already presented argument at the beginning of this analysis, topics revolve around specific types of food. We find frequent words about specific products. For example:
  - Topic1: Sandwich, cheese, bread, chicken, coffee, …
  - Topic2: Steak, wine, salad, sauce, …
  - Topic3: Chinese, rise, noodles, salad, sauce, …
  - Topic4: Pizza, toppings, pasta, wine, olive, …
  - Topic5: Mexican, tacos, guacamole, burritos, chili, …
  - Topic6: Burger, cheese, fries, sauce, onion, …
  - Topic8: Japanese, sushi, rolls, tuna, tempura, …
  - Topic10: Indian, pita, curry, tofu, rice, …
- We also identify how similar types of cuisine are grouped into the same topics. For example:
  - Pizza and Italian. Topic4
  - Chinese, Asian and Vietnamese. Topic3
  - Japanese and Korean. Topic8
  - Thai and Greek. Topic10

- Finally, we identify which are the most frequent terms for all topics; common remarks used when making an assessment, regardless the type of cuisine. Here, besides the identified words ***service*** and ***time*** at the beginning of this analysis, we find the following frequent words:
    - ***Quality***
    - **Kind words** like: *love, nice, friendly, awesome, pretty, …*
    - **Numbers**. We can check that all topics appear numbers (waiting time for a service?)

2.2. Topics. Positive Vs negative ratings

Approaching a second part of the analysis, we study the **similarities and differences between positive and negative assessments**. We do a previous analysis to understand which valuations may be considered positive and which negative, as well as if there is a data subset that results more significant.

For each city, we obtain the **number of reviews per rating** assigned (number of stars, from 1 to 5). We show this data for the 10 cities which have more reviews. It seems that the proportion of negative ratings (1 and 2) is tiny compared to positive ratings (4 and 5), even if we consider rating 3 as negative, it does not reach 50%. We also decided to obtain these standardized data as a **percentage** of the total valuations of each city and confirm that valuations 1, 2 and 3 are well below 50% for the 10 cities with more reviews.

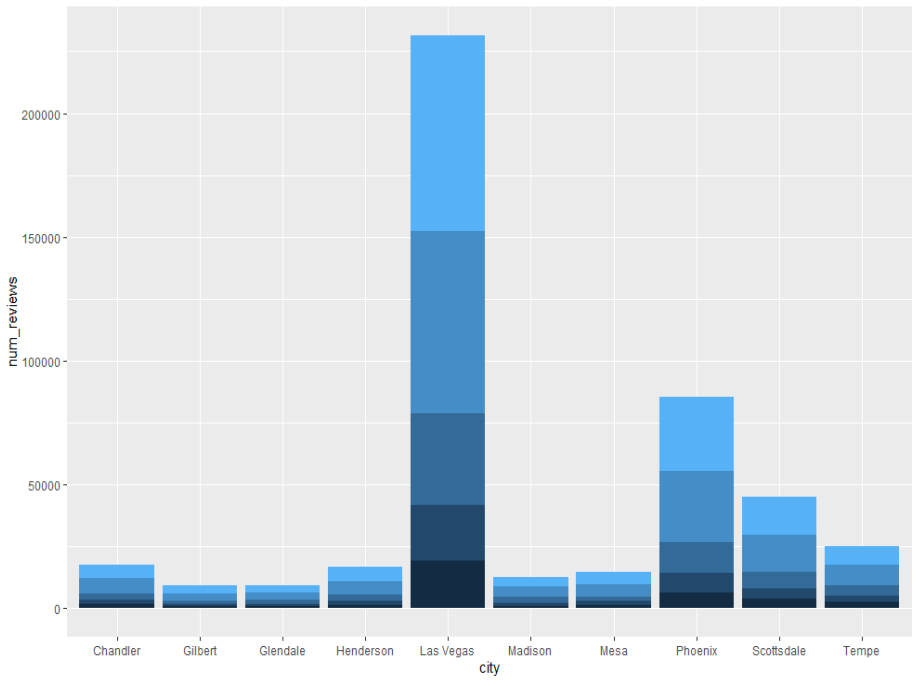


Figure 12. Number of reviews per rating. Top-10 cities per review

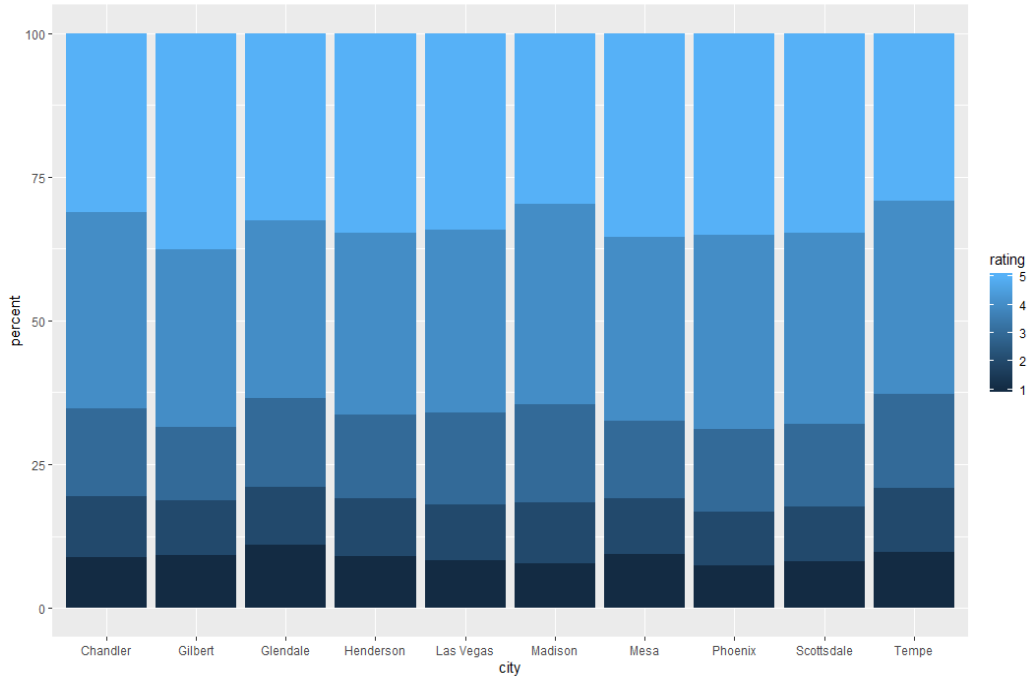


Figure 13. Percentage of reviews per rating. Top-10 cities per review

This leads us to conclude:

- We must consider as **negative 1, 2 and 3**

- The valuations of this social network tend not to be overcritical.

- The proportion is quite similar for the 10 cities with more reviews, so let's choose **"Las Vegas" as a subset, being the one with most reviews**.

I apply the same **LDA model** for the **identification of topics** that I used in the previous section, with the particularities:

- I apply **stop words** to remove irrelevant common words for this analysis.

- I search **5 topics for positive ratings (4 and 5) and 5 topics for negative ratings (1, 2 and 3)**

- I use the graph () function to create a graph that shows the **15 most repeated words** of the respective topics, distinguishing each topic with a color, and using the opacity of the node to indicate the weight of that word in the topic. The type of chart is hierarchical and the layout circular.

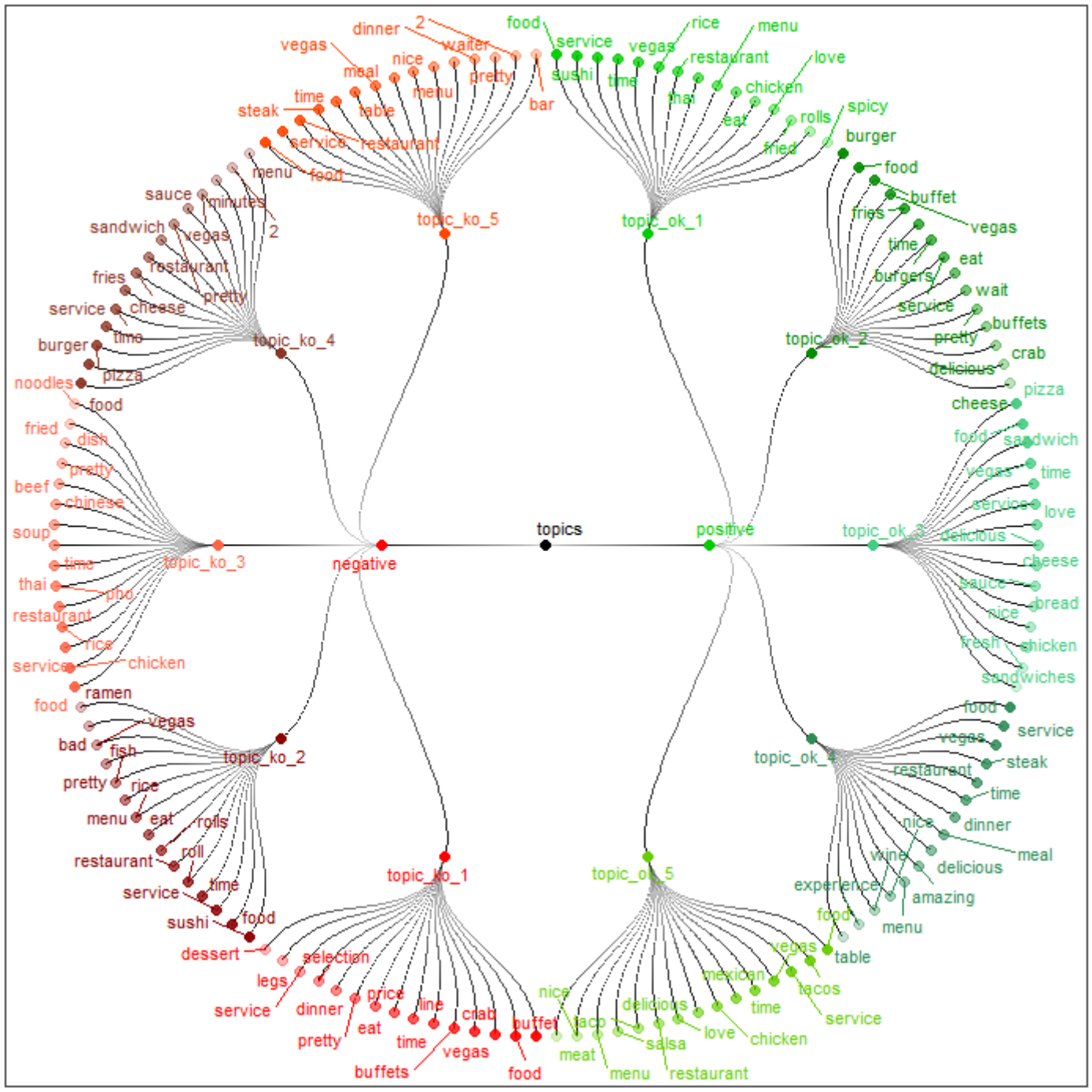


Figure 14. Topics YELP for "*Las Vegas". Positive Vs. negative ratings. LDA topic model*

As done for the previous section, I use the **word clouds** for the **5 positive and negative topics** to enable a deeper analysis.

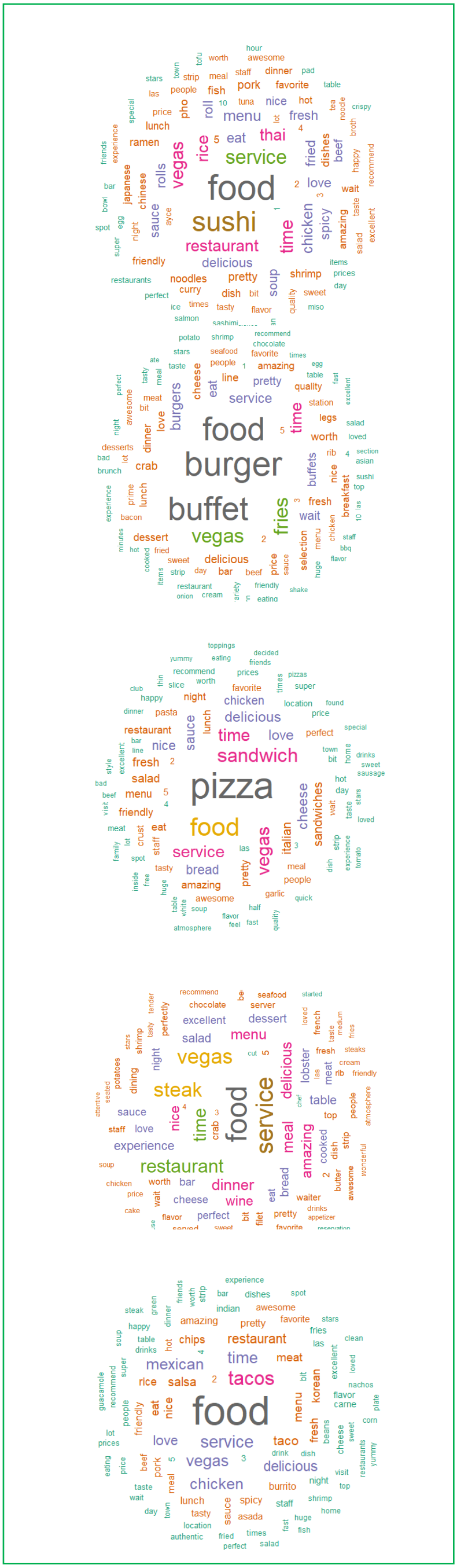


Figure 15. Topics "Las Vegas" by rating. Positive vs. negative ratings

Confronting positive topics versus negative ones we observe:

- Like what happened in the previous section, the topics are grouped by types of cuisine (or similar types of cuisine). In this case, it is more mixed since we have identified only 5 topics, and not 10.

- Certain words are repeated, both for positive and negative assessments. E.g.: **service, time, quality, price**, ...

- There are very few specific words containing negative connotations on the negative reviews. The only word found with true relevance is "**bad**".

- We identify the word "**minutes**" as relevant and repetitive on different topics of negative assessment.

To provide context to those identified words, I analyze all the reviews by tri-grams (groups of three adjacent words), applying stop_words. I extract the most repetitive tri_grams containing the words we want to analyze, for both positive and negative ratings, by applying stop_words. As seen in the following graphs, the 10 most repetitive tri_grams on positive reviews and the 10 on the negative ones. The X axis shows the number of repetitions. For negative reviews, I have multiplied this value by -1 for a better display of each subset.

- **Quality:** We clearly appreciate negative and positive connotations in each sub-set

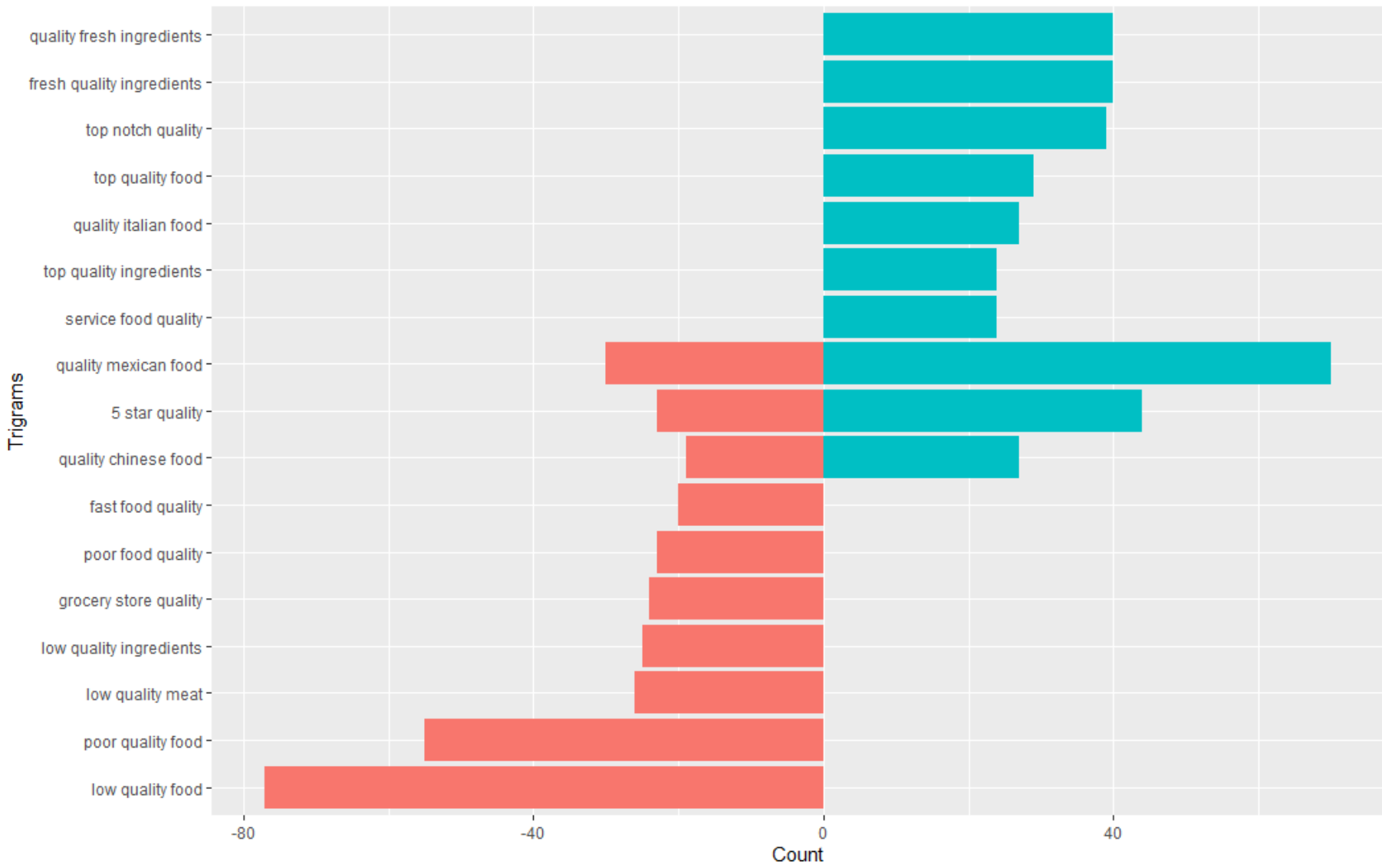


Figure 16. Top 10 trigrams with the word: QUALITY. Positive Vs. negative ratings

- **Service:** here once again, we also clearly recognize positive and negative connotations for each subset. As the concept "customer service" is very often repeated, we also decided to search for the word "customer". We can observe that in the case of "**customer service**", positive connotations are more generic and flattery, and in the negative ones, we can find specific concepts such as "customer service training", "customer service skills", “customer service experience”,...

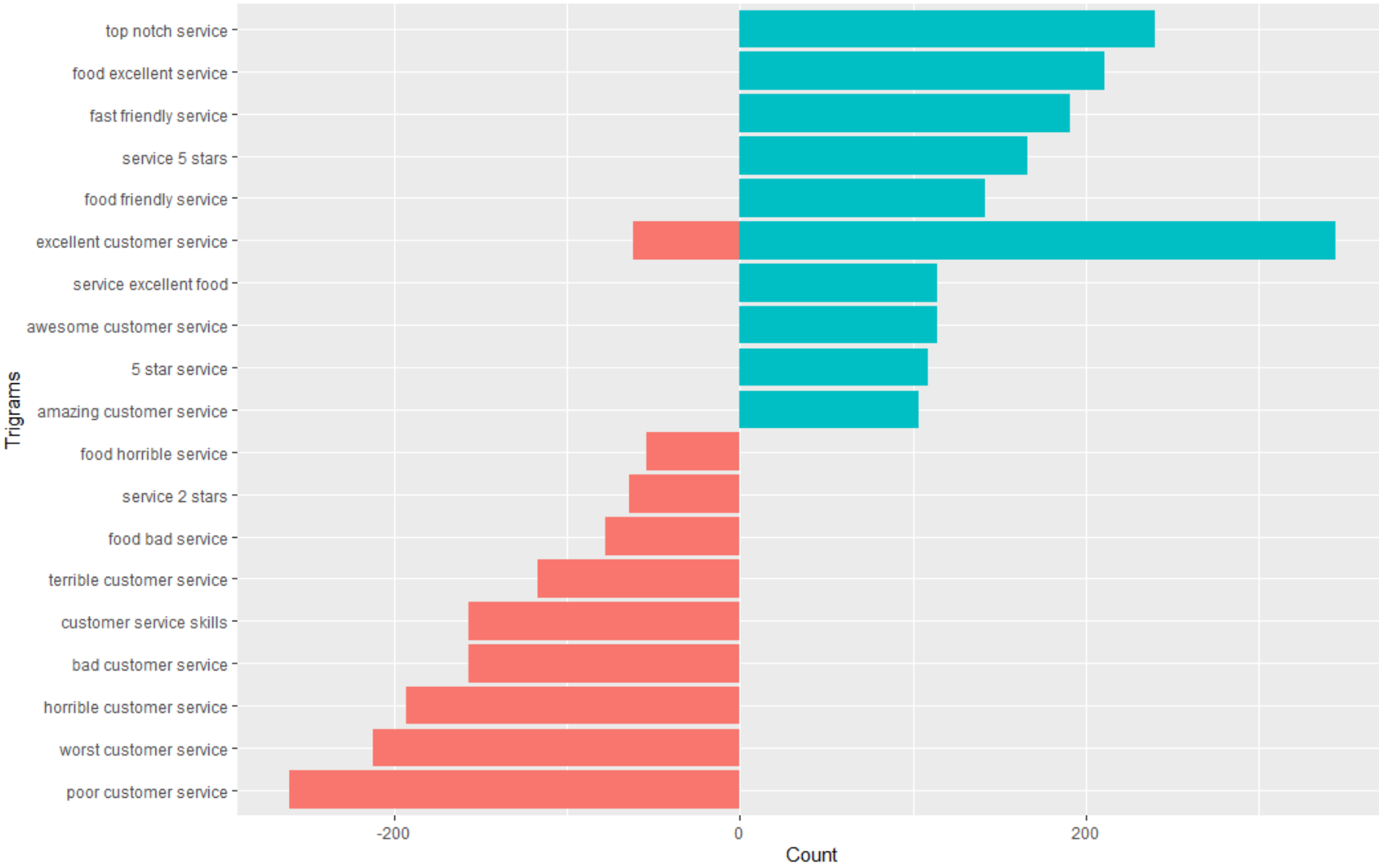


Figure 17. Top 10 trigrams with the word: SERVICE. Positive Vs. negative ratings

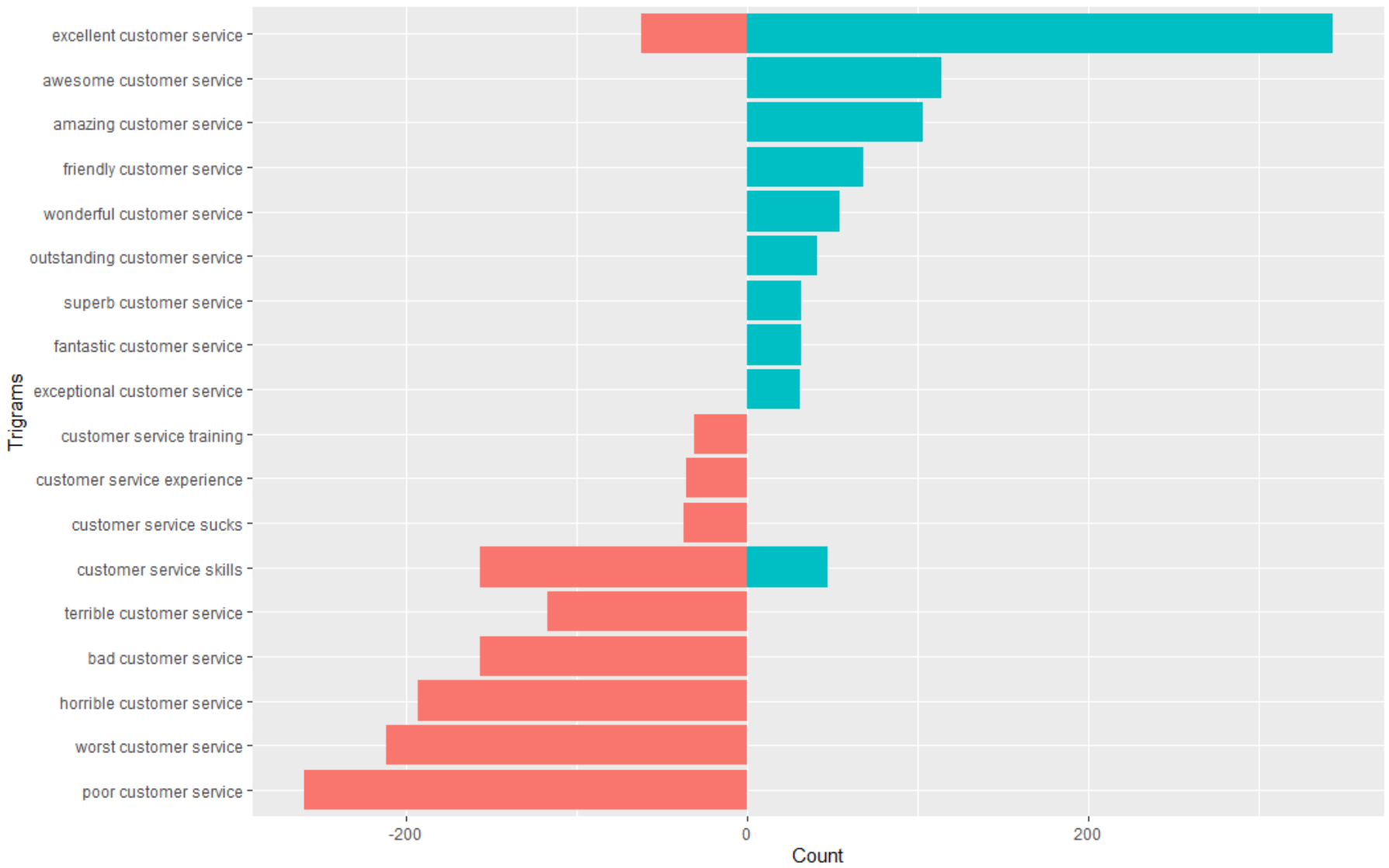


Figure 18. Top 10 trigrams with the word: CUSTOMER. Positive Vs. negative ratings

3 Cuisine clustering and map construction

In this section, I do an analysis in order to identify similarities between different types of cuisine. For this purpose,

I explore two main ways of analysis:

- **Conversations**. I analyze the reviews, identifying similarities in the digital conversations about the different types of cuisine.
- **Customer preferences.** Although is not expressly requested in the exercise, I find interesting to also analyze the customer preferences regarding the recurrence in visits to different restaurants, by using clustering techniques to find common points.

3.1. Similarities base on the features

For the made of the Cuisine Map, I chose the following types of cuisine subset, for being the one with more reviews and relevance:

"Asian Fusion", "Buffets", "Burgers", "Chinese", "Delis", "Filipino", "French", "Indian", "Irish", "Italian", "Japanese", "Korean", "Latin American", "Mediterranean", "Mexican", "Pizza", "Sandwiches", "Seafood", "Steakhouses", "Sushi Bars", "Thai", "Vegetarian", "Vietnamese"

I get the following similarity matrix using all the reviews. Initially, I get the no IDF analysis results, by using Jaccard and Cosine algorithms for the distance calculation.

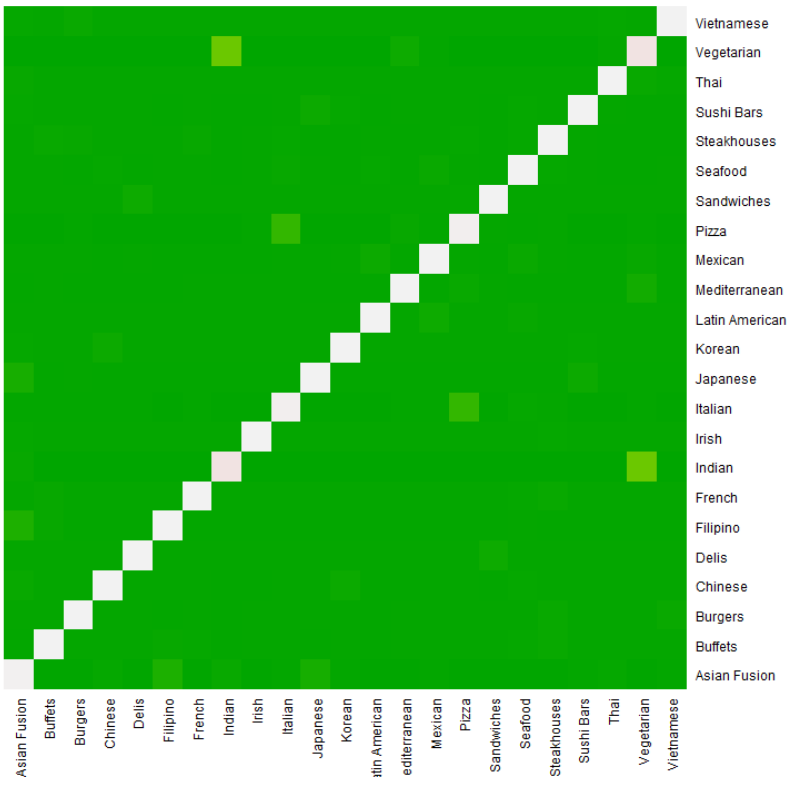


Figure 1. Similarity Matrix. Cosine

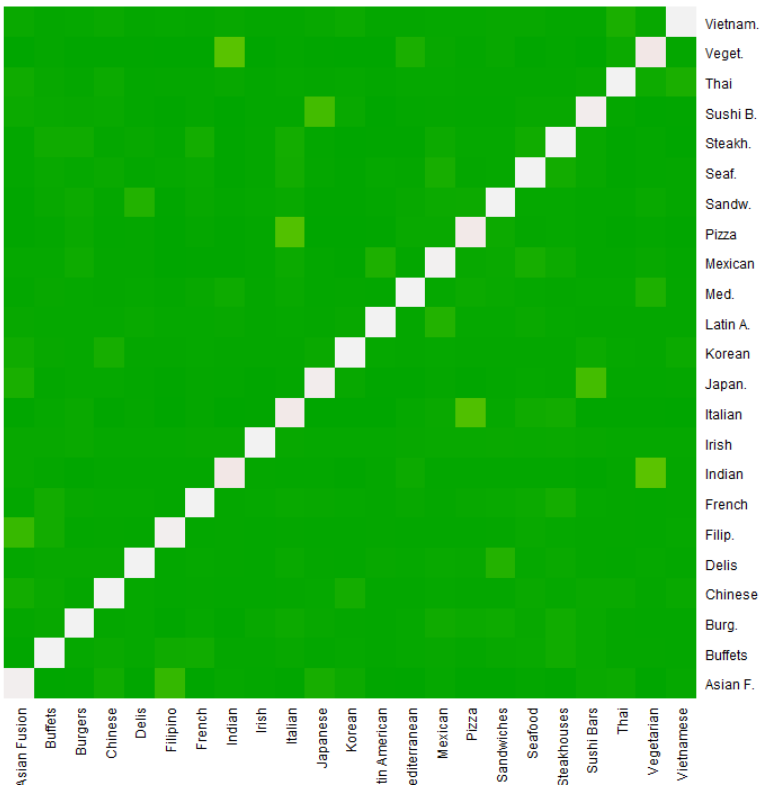


Figure 2. Similarity Matrix. Jaccard

In this analyzed case, we can appreciate that the Jaccard algorithm works better than Cosine. More nuances and similarities can be distinguished. The relation between **Vegetarian and Indian**, as well as between **Italian and Pizza**, are still highlighted in both, but with the Cosine algorithm we can see other similarities such as:

- **Sushi Bar, Japanese**
- **Vegetarian, Mediterranean**
- **Filippo, Asian Fusion**

In order to improve our Cuisine Map and refine the similarities map, I use two techniques:

- **Similarity Matrix from TF-IDF**. TF to penalize those very frequent terms in each cuisine type, and IDF to penalize those frequent ones between various cuisine types.
- **Similarity Matrix from LDA.** Based on the terms by topic distribution

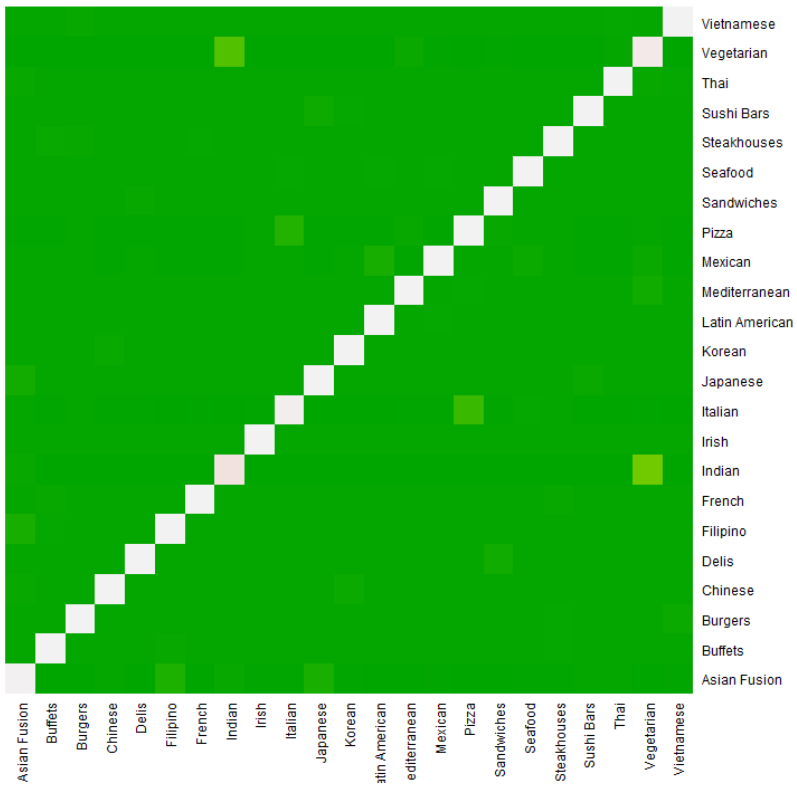


Figure 3. Similarity Matrix. TF-IDF

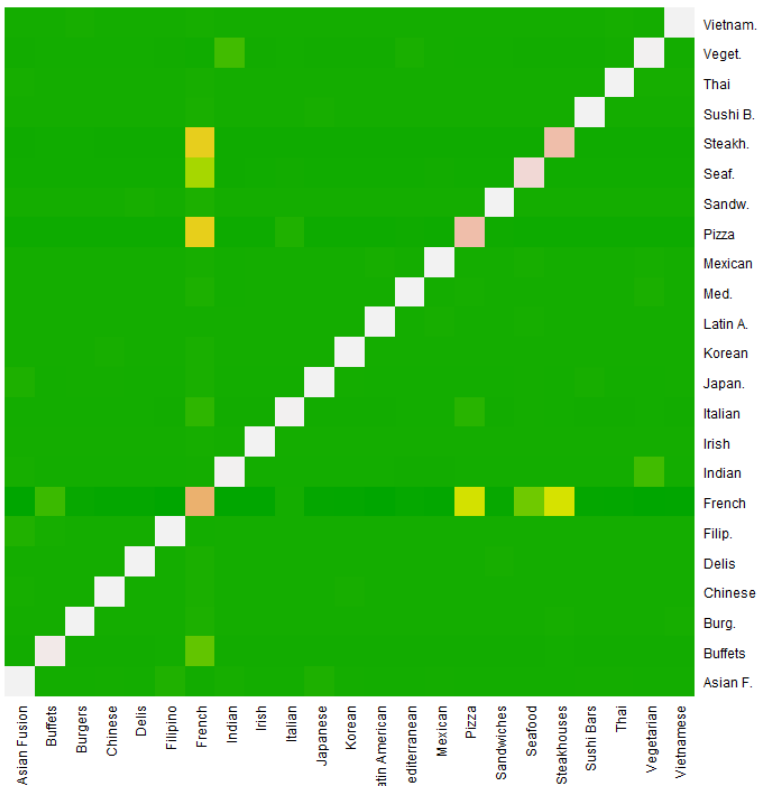


Figure 4. Similarity Matrix. LDA

If we only apply TF-IDF, we do not get much improvement, but by also applying LDA we realize that we are able to identify additional nuances to those already identified in the previous section, such as the similarity group **Steakhouse, Seafood y Pizza**.

To make progress in the Cuisine Map identification of similitudes, I used the clustering techniques. Specifically, the **K-MEANS** algorithm.

Clustering of the Cuisine Map. Jaccard distance. We apply the clustering K-MEANS algorithm to the Cuisine Map we obtained with the Jaccard distance calculation. Next, you find the obtained display for the 4 clusters and for 6.

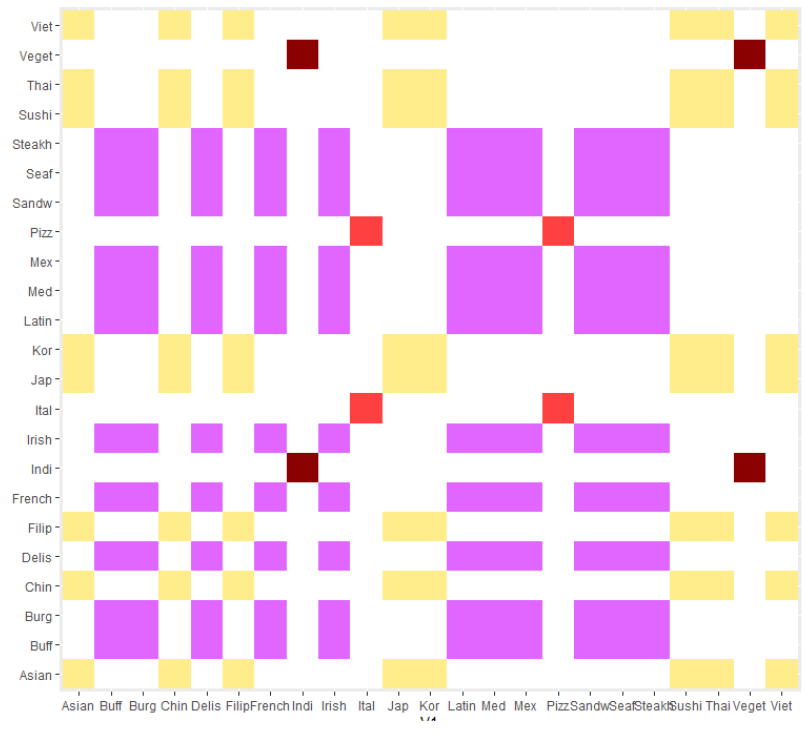


Figure 5. Similarity Matrix. Jaccard + K_Means. 4 clusters

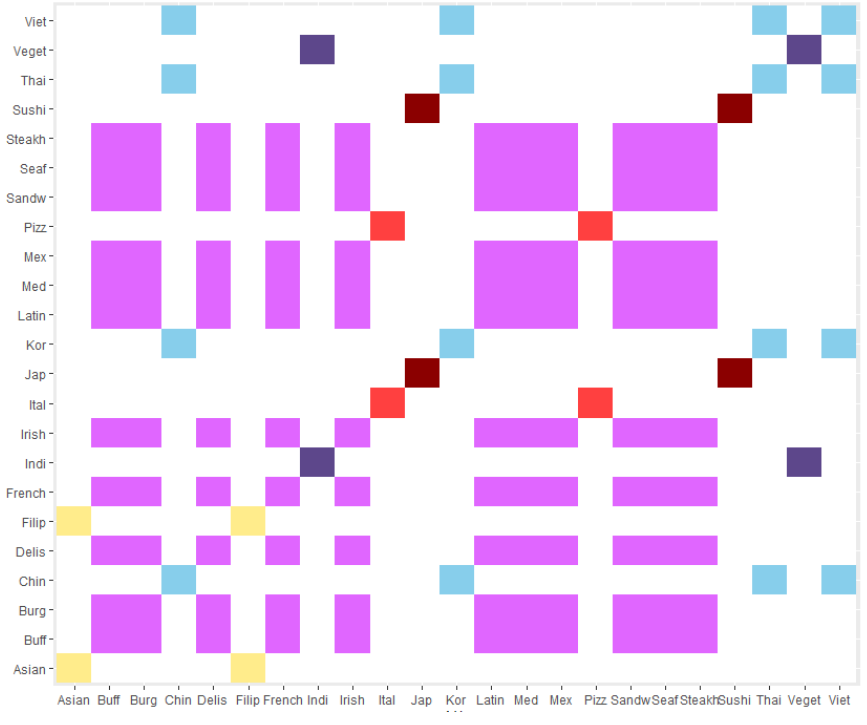


Figure 6. Similarity Matrix. Jaccard + K_Means. 6 clusters

Here, we can appreciate well defined the cuisine types similarities among themselves. Some match with the already identified in the above sections:

- **Pizza, Italian**
- **Vegetarian, Indian**

We also identify other groups perfectly distinguished. For the case of 4 clusters:

- Asian style types of cuisine: **Vietnamese, Thai, Sushi Bar, Korean, Japanese, Filipino, Chinese, Asian Fusion**
- Other types of cuisine: **Steakhouse, Seafood, Sandwich, Mexican, Mediterranean, Latin American, Irish, French**

For the 6 clusters case, we further concretize the Asian styles, differentiating the specific clusters:

- **Sushi Bar, Japanese**
- **Filipino, Asian Fusion**

- **Clustering of the Cuisine Map. LDA**. We use the clustering K-MEANS algorithm to the Cuisine Map that we got by using the identification method of LDA topics. Next, we show the obtained views for the 4 clusters and for 6.

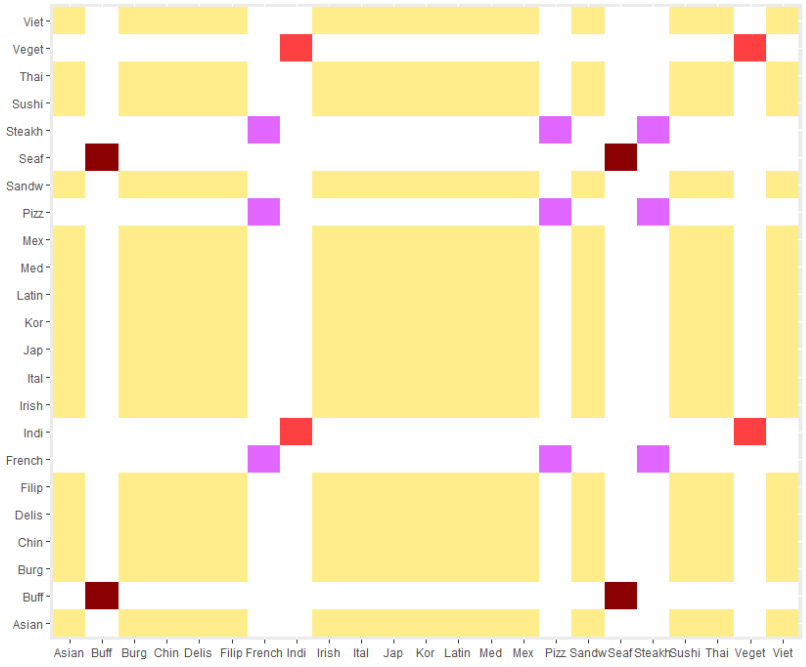


Figure 5. Similarity Matrix. LDA + K_Means. 4 clusters

Figure 6. Similarity Matrix. LDA + K_Means. 6 clusters

For the clustering analysis case, with 4 clusters, we get a too generalist differentiation, distinguishing only:

- **Seafood, Buffet**
- **Steakhouse, Pizza, French**
- **Vegetarian, Indian**

The 6 clusters analysis does get to also identify interesting groups such as:

- **Japanese, Filipino, Asian Fusion**
- **Sandwich, Delis**

3.2. Similarities base on the customer preferences

I do an analysis of the frequency of visits to restaurants of each type of cuisine, by using the customer reviews. I just analyze the customer subset with more than 10 reviews. By applying the clustering K-MEANS algorithm, I get those views for 4 and 6 clusters respectively.

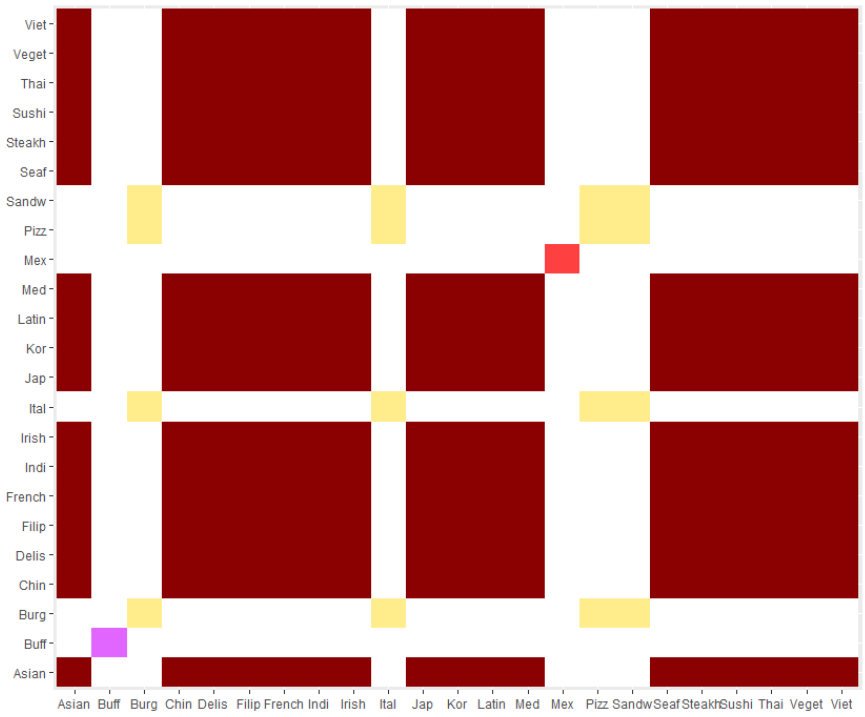


Figure 7. Similarity Matrix. Visits frequency. K_Means. 4 clusters

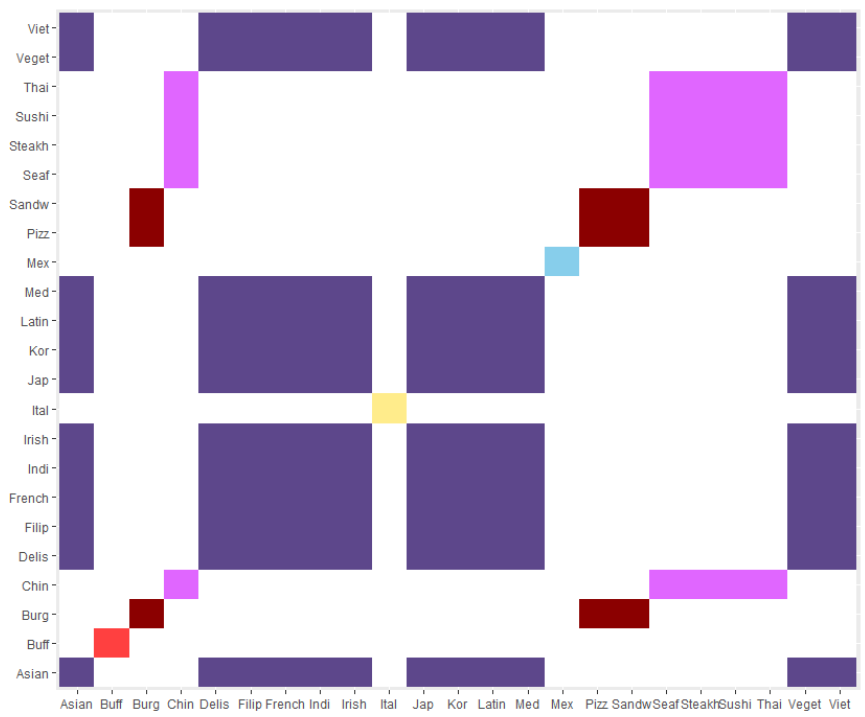


Figure 8. Similarity Matrix. Visits frequency. K_Means. 6 clusters

In this case the 4 clusters analysis is very generalist and the one with 6 gets to affine more groups of similarities. For both cases, the identified groups are very different to the ones we got in the analysis by conversations. For the case of 4 clusters identifies as independent clusters of the generic group:

- **Sandwich, Pizza, Italian, Burger**
- **Mexico**
- **Buffet**

The 6 clusters analysis also identifies:

- **Thai, Sushi Bar, Seafood, Steakhouse, Chinese**
- **Italian** (remaining in an independent group **Sandwich, Pizza, Burger**)

I have found it very interesting to see how different clustering techniques and algorithms have allowed me to refine the similarities identification between different types of cuisine. Also, **I have been able to clearly see the differences between similarities that identify the conversations analysis and the customers' preferences o frequency of visits**. By looking at the results**, for the first case**, the similitudes have more relation with the **type of cuisine referring to the cuisine style, ingredients or tastes.** And in the second I get to identify **a segmentation more influenced** by parameters, such as age and specific customer **preferences**. For instance, the subset Sandwich, Pizza and Burger is more typical of teenagers and informal customer types. The Thai, Sushi Bar, Seafood, Steakhouse and Chinese subset is more related towards a segment of varied and informal food, and we also identify in independent clusters, the Italian, Mexican and Buffet's type of food, which are the kind of restaurants that respond to very specific and marked customer's preferences.

4 Dish recognition

The purpose of this section is being able to build a dish recognizer that allows us to discover the most common and popular dishes among the different types of cuisine. To do so, I have explored the word2vec implementation in R, since both TopMine and SegPhrase have caused me numerous problems of execution on my Windows system.

Among the given choices of cuisine types, I pick the Italian one, since it is one of my favorites and, I know it well, since I travel to Italy very often. Hoping not only to delve into an environment that I like, and it seems familiar to me, but also to make me discover new Italian dishes to try and taste!

4.1. Manual tagging

I change a bit the given instruction for this exercise, since for the used tool word2vec, we do not need as an input, a list of positive and negative dishes, but a list, as complete as possible, of popular Italian dishes, that allow the algorithm to apply word association techniques (mutual information), and to extract other words very similar to those indicated. If we manage to do well this entry list, as well as training properly the algorithm with our data-set, we shall get more Italian dishes regularly commented in our reviews.

I check the given list to carry out the exercise and:

- I cancel the false positives
- I change the false negatives into positives
- I finally cancel the rest of negatives

4.2. Mining additional dish names

As already introduced in the previous paragraphs, I use the R WORLD2VEC PACKAGE. This package finds similarities of words and/or phrases in a corpus by using the association techniques such as mutual information.

Once the package is installed, I use the prep_word2vec function to prepare the file with the corpus that helps me to create the word association model. This package uses the tokenizer function to clean and apply the lower case on the text or corpus and get the n-grams ready to allow us to model the words association, not just on the independent words, but also on consecutive group of words. I consider this fundamental, since most popular dishes are made by more of one word. A three level for the n-grams it seems more accurate and I indicated as a parameter of the prep_word2vec function.

Next, I “train” the already prepared corpus in the previous step. We get this by using the train_word2vec function that creates the word association model. The model size may be parameterized with the vector’s parameter. I create the model with a 200 vectors dimension, with the purpose of achieving a very accurate model. The assigned parameters to the train_word2vec function are the following:

```
vectors = 200

threads = 4

window = 12

iter = 5

negative_samples = 0
```

However, indicate that I have been cautious with the iter parameter assignment (number of iterations and reading made on the corpus), since I am working on a million words. I start with this parameter in a controlled way, and if I notice that the results are not of quality enough, and introduce too much noise, I will gradually upload it.

Once I have created my association model, I do the following tests:

1.- Simple test

I start by getting the 10 closest words to “spaghetti”. The result is as follows:

1	spaghetti	1.0000000
2	meatballs	0.7746775
3	meat_sauce	0.7181702
4	meatball	0.6599619
5	spagetti	0.6454983
6	meat_balls	0.6374400
7	marinara	0.6171008
8	kobe_meatballs	0.5788949
9	kobe_beef	0.5716422
10	pasta	0.5709684

It is not a bad result for a simple test. We can observe many concepts relative to Italian dishes. But also, it is true that we can better this result, since there are too many repetitions.

2.- Test with the reviewed Italian dishes list.

For this case, I used the exercise given list, reviewed and improved by following the explained indications in the previous section. In total, I have 109 confirmed popular dishes. I get the closest 50 words (or groups of words up to 3) to this set of dishes. The obtained result is:

1	coulis	0.7597256
2	pumpkin_seeds	0.7589192
3	croquette	0.7509706
4	marscapone_cheese	0.7494074
5	lump_crab	0.7409292
6	asparagus_spears	0.7406733
7	marscarpone	0.7381971
8	fraiche	0.7349806
9	fruit_compote	0.7322817
10	candied_hazelnuts	0.7320898
11	compote	0.7313422
12	bitter_greens	0.7298029
13	creamed	0.7283605
14	sauteed_vegetables	0.7250014
15	raspberry_sorbet	0.7221456
16	creme_fraiche	0.7215438
17	maple_syrup	0.7209150
18	ice_creams	0.7199504
19	smores	0.7176320
20	sugar_coated	0.7162466

In this case, almost all the results correspond to typical Italian dishes. I consider then that the precision level of the model is quite high.

I decide to explore a bit more the created model limits, and I search for the closest 12 words “dessert”. This is the result:

1	tiramisu	0.6451595
2	cheesecake	0.5974043
3	desert	0.5861063
4	cookies	0.5829564
5	creme_brulee	0.5786894
6	pastas	0.5732109
7	tirimisu	0.5687358
8	chocolate_cake	0.5589650
9	cannoli	0.5581950
10	chocolate_mousse	0.5510865
11	profiteroles	0.5481244
12	pumpkin_cheesecake	0.5472883

A quiet surprising result and with heaps of possibilities of application and use cases.

The obtained results with the word2vec package have been very surprising. This association model is created based on the existing context between two words. This means, two words are closer if the words that surround those are similar. In the case in question, finding common dishes, is a model that really works well, since when we talk about a cuisine dish, we normally do it in a very specific context.

Hence, I would add that I believe this model has really work well thanks to the context of our corpus or text, which was very limited (only Italian restaurants reviews). Surely this is one of the most important keys that make this model work well, the bounded of the context to be modeled.

5 Popular dishes and restaurant recommendation

The aim of this capstone module is to build a recommender system for typical dishes; this is, a system that allows you to obtain a typical and popular dish ranking of a specific type of cuisine, identifying which ones are more appreciated and which less. To do so, the restaurants reviews data for this type of cuisine must be mined, applying sentiment analysis techniques and modeling what elements influence in whether a dish is popular or not, and therefore, in our ranking.

Furthermore, in a second section, we will build a restaurant recommender system for a specific dish. By doing this, we can help people who want to try a new type of cuisine, by recommending what to try and where.

I pick the Italian cuisine to carry out this module's tasks, since this is a cuisine that I love and know well. In the other hand, I have applied R to mine the reviews and build my ranking model.

5.1. Mining popular dishes

We must design a ranking model to build our recommender system. To achieve it, we ask ourselves which are the elements that influence in making a dish popular or typical, and therefore it becomes chosen or not as the favorite to try. I believe there are three scopes we must analyze:

- **Popularity**. How much do they talk about this dish?
- **Perception**. Do they talk well or bad about it?
- **Specialization**. Is it a common dish or is only served in very specific places?

Sentiment analysis: Sentimentr package

For the sentiment analysis, I decide to explore a tool or specific package for the sentiment detection, through the review's analysis. I could have only considered the valuation with which the user cataloged his comment (stars or review score), but I have decided to bet on a more sophisticated ranking model, since the specific perception about a dish may differ from the global experience opinion that took place in the restaurant.

I decide to explore the Sentimentr package of R. I fundamentally chose this option for two reasons:

- This package allows to mine the texts, by using tokenization in phrases. In this way I can focus much more on the sentiment analysis of the specific dish done comment.
- Sentimentr is a considerably evolved package, since it takes in consideration how some commonly used "valence shifters" influence in our conversations. Those "valance shifters" amplify, nuance and even completely change the meaning of our words. Classifying those into:
    - **Negator**. Flips de sign of a polarized word (e.g., "I do **not** like it")
    - **Amplifier**. Increase the impact of a polarized word (e.g., "I **really** like it")
    - **De-amplifier**. Reduce the impact of a polarized word (e.g., "I **hardly** like it")
    - **Adversative**. An adversative conjunction overrules the previous clause containing a polarized word (e.g., "I like it **but** it's not worth it")

After applying the sentiment analysis functions of the Sentimentr package, I review some of the obtained **results** to check how is working. As an example, I show the ten phrases related to the "gorgonzola" word, which obtain the highest score in the sentiment analysis, and the ten with the lowest score.

“Gorgonzola” highest score:

sentiment	sentence
0.7349339	all the cheeses were good, but the gorgonzola was my favorite.
0.7361216	pro's - lunch special good deal (large personal pie & salad) - gorgonzola salad fantastic!
0.7639872	and, as much as i love green chile, my new fave is the fig and gorgonzola app.
0.7688539	the good news was that the gorgonzola sauce was very tasty.
0.7914843	they also have great salads and some really interesting flavors of gelato, including chocolate chile, pear gorgonzola, and more.
0.8617864	i love oreganos, but i also love gorgonzola.
0.8634629	actually, i pretty much love oregano's for their aunt shirley's gorgonzola pizza an order of their nice, meaty wings with mixed sauce.
0.8874480	they weren't bad, but the fresh bread dipped in the gorgonzola sauce was making us smile more than the potato dumplings were.
1.0509126	the steak was very tender and well-seasoned with balsamic and the pasta was very good & cheesy with small chunks of gorgonzola throughout..
1.1313708	their salmon carpacio is really good and i love love love their panini with grilled gorgonzola and figs.

“Gorgonzola” lowest score:

sentiment	sentence
-0.25298221	some bites you may find the gorgonzola to be overpowering.
-0.25819889	it has mandarin orange pieces, pear, arugula, walnuts, endive, gorgonzola cheese, radicchio with lemon dressing.
-0.27950850	too much for 2 people after the wonderful medium sized pairing of the pear/gorgonzola pizza and the chicken/feta pizza.
-0.29121760	the sauce was good, but the gorgonzola overpowered the flavor of the steak.
-0.31980107	the triple cream brie was almost undetectable in flavor and the gorgonzola, although more pronounced, didn't seem like it was high quality...
-0.32422986	he substituted the porterhouse at no extra charge with the same gorgonzola sauce that would have been on the ribeye.
-0.33407655	i got the corned beef hash with fried polenta and gorgonzola, and poached eggs.
-0.35178118	my sister got the gorgonzola pear salad and after having a bite i wished i would have ordered that for lunch tomorrow!!
-0.37500000	the cranberry walnut chicken salad, our old fave, lacked gorgonzola and had way too much celery.
-0.42640143	flaky, oh-so-thin crust, topped with gorgonzola cheese, tomato, bacon, and arugula, this meal of goodness leaves no room for leftovers.

-0.43301270	gorgonzola is evil.

Those shown results are very similar to others I have analyzed, where the positive ratings usually correspond to nearly the 100%, but for the negative ones we find more inaccurate results (in red and yellow in the shown example). I think the results would be much better by updating the dictionaries (polarity or valence shifter), to adapt those to the specific contexts of this social web. Nevertheless, since the obtained result is quite acceptable, I decide to continue with the ranking model, and go on with those improvements later.

Ranking system modeling

We move on to modeling each one of those concepts:

1.- Popularity: Getting an indicator that allows us to value this aspect is very simple. We only need to count the number of occurrences or times the dish gets mentioned among the different reviews. We also calculate it normalized with the purpose of making it comparable with other aspects to be evaluated. To do so we divide it by the total number of Italian restaurants.

Popularity(d) = Sum(Occurrences(d)) / TotalRest

2.- Perception: To evaluate the way a dish is commented, how good or bad, we must do a sentiment analysis of the conversations. Later we explain what sentiment analysis algorithm we chose for this exercise. Besides this valuation, we think it also influences, and therefore we must consider it, if a specific review has got positive valuation scores. Specifically, the votes$useful indicator must work as an amplifier of the reflected feeling, whether is positive or negative. Since our sentiment indicator will be in the range (-1, 1) we will use the following formula that applies this amplifier effect, that prevents very high values from distorting our comparison:

*Perception(d) = Mean (Sentiment(rew) * (1 + log (1 + #VotesUseful(rew)))*

3.- Specialization: we evaluate a dish specialization by seeing how many restaurants have reviews about it. The fewer restaurants are, the greater the specialization is. This indicator should be valued based on the reviews that the analyzed dish has obtained, since it only becomes significant when the number of reviews is relevant. We model the specialization indicator with the following formula, similar to the one that models the IDF calculation (Inverse Document Frequency):

Specialization(d) = log ((Sum(Occurrence(d)) + 1) / DistinctRest(d))

With these three indicators, our ranking model remains as follows:

Ranking(d) = **α** * *Sum(Occurrences(d)) / TotalRest* + **β** * *Mean (Sentiment(rew) * (1 + log (1 + #VotesUseful(rew))))* + **μ** * *log ((Sum(Occurrence(d)) + 1) / DistinctRest(d))*

Popularity(p) Perception(p) Specialization(p)

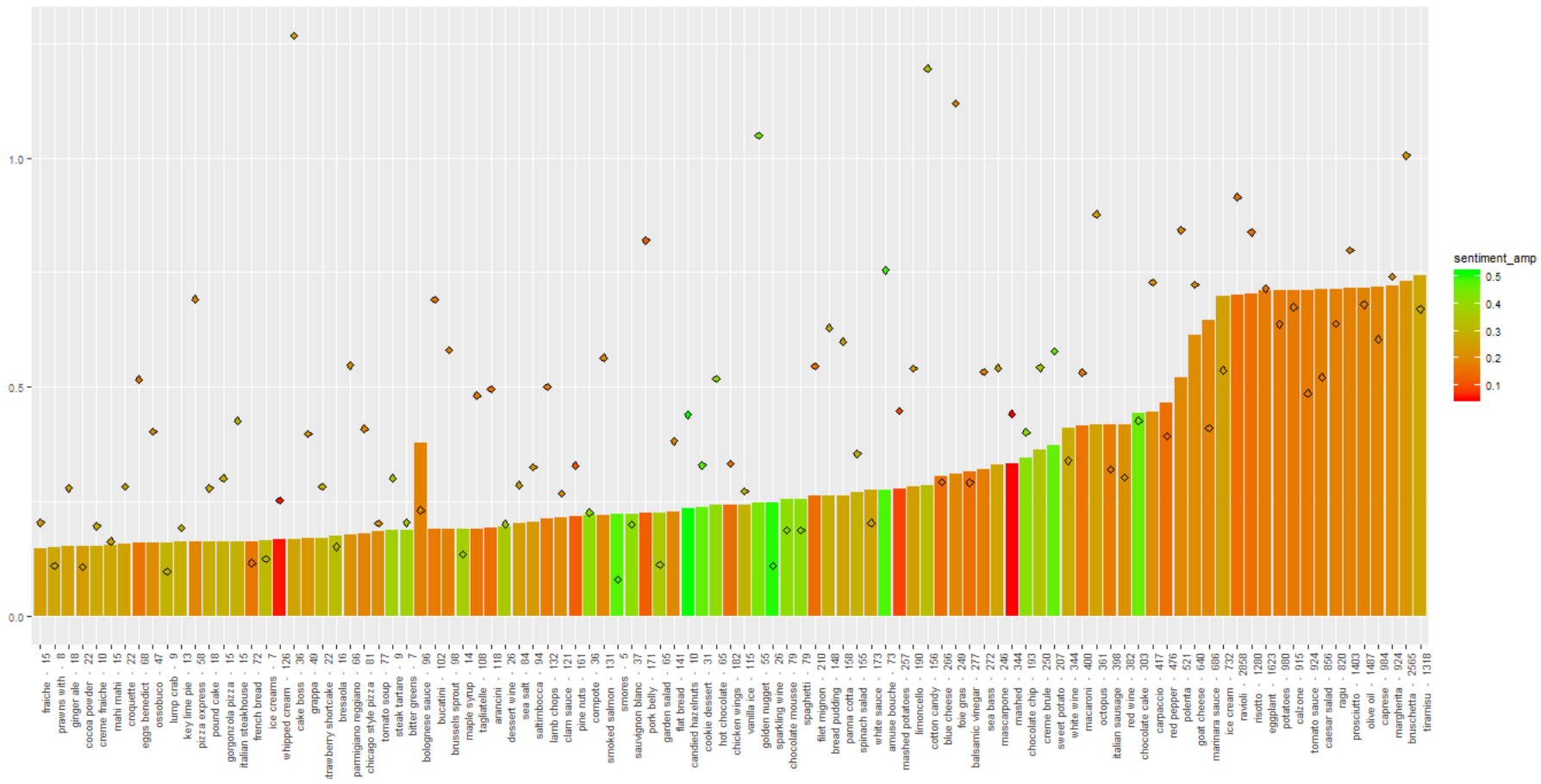
After several tests, the most consistent proportion for the ranking is:

α=0.6

β=0.35

μ=0.05

I get the following graph that reflects the sentiment analysis (amplified according to the received scores and to the criteria defined in our model, which would be the ranking), and I mark with a rhombus the level of “specialization” of each dish.



5.2. Restaurant recommendation

In this second section we must build a restaurant recommendation system, for a specific given dish that a user has decided to try. For this case, I think the sentiment analysis of the model must weight much more. I use a similar formula to the one in the previous section, regarding sentiment analysis. But in this case I will calculate by dish and restaurant. I will add another concept to reflect the *popularity* of the restaurant respecting this dish. In this case is the number of times the dish is mentioned in the reviews, with respect to the total number of reviews than mention it.

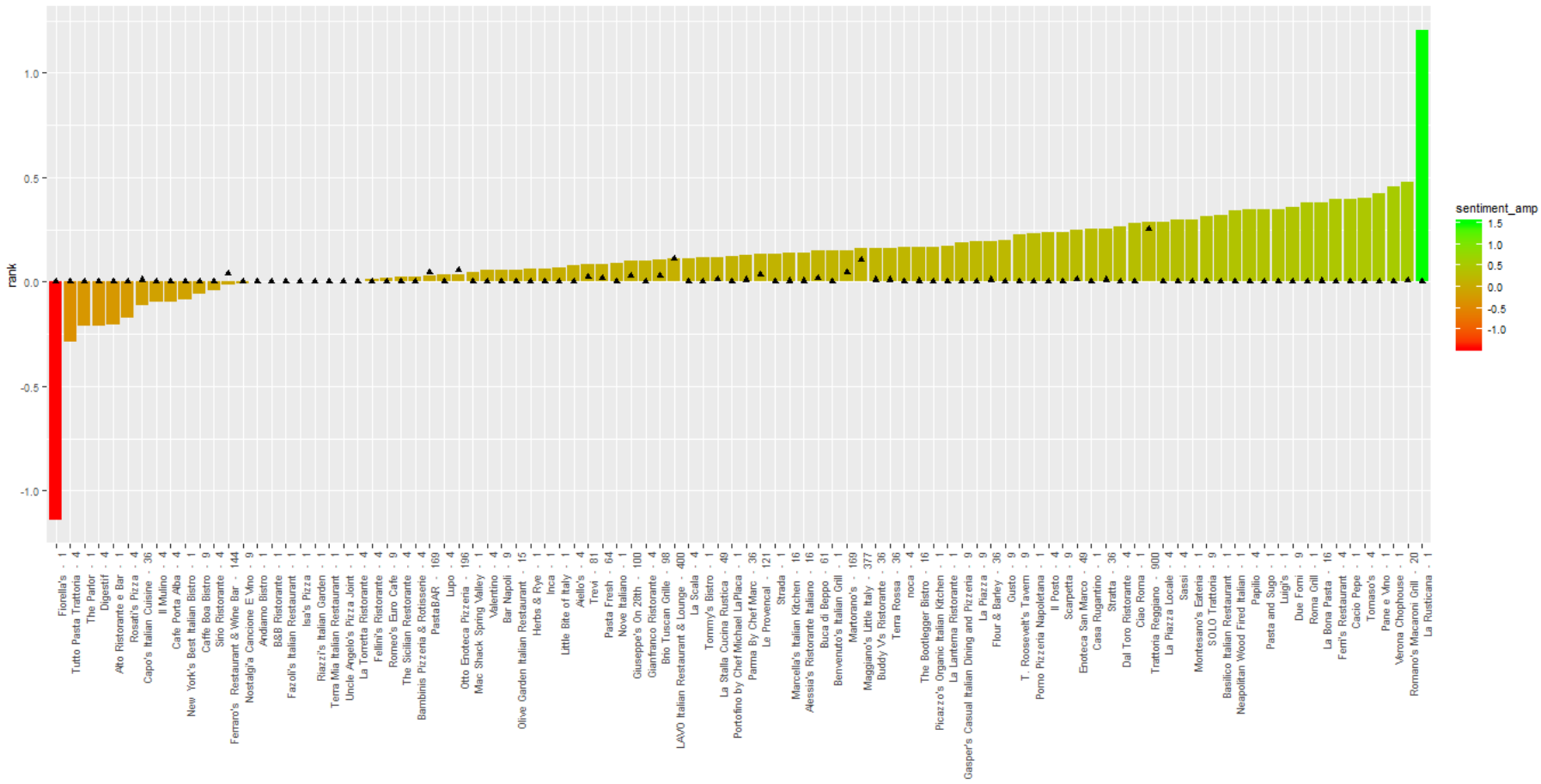
Ranking(d, rest) = **α** * *Sum(Occurrence(d, rest)) / Sum(Occurrence (d))* + **β** * *Mean (Sentiment(rew, d. rest) * (1 + log (1 + #VotesUseful(rew, d, rest)))*

Specialization(p, rest) — Perception(p, rest.)

α=0.2

β=0.8

Next I include the ranking visualization of our model, using as an example the result of searching for restaurants to eat "*spaghetti carbonara"*. Next to the name of the restaurant, the number of occurrences is indicated, and we mark the level of specialization of the restaurant with a symbol:



6 Predictive model. Public health inspections

In this course challenge section, I explore different analysis strategies to predict which of the provided data set restaurants would pass the public health inspection test. To get this, I have a training data set, which in addition to the data provided below, it has a label column that indicates whether the mentioned test has passed. These are the data:

- Restaurant reviews
- Cuisine type
- Zip code
- Average rating
- Number of reviews

I specify the different ways of analysis that I have followed. I have explored, using the R ***mlr*** package, the following predictive modelling algorithms:

- **Logistic Regression**
- **Decision Tree**
- **Random Forest**
- **Xgboost**

On the other hand, I have analyzed several strategies for variables classification:

- **Unigram and bigram** analysis. TF-IDF.

Unigram and bi-gram analysis, applying **Mutual Information** to focus on the words closest to the ones related to cleaning and hygiene issues.

6.1 Predictive model

After importing the provided data, I prepare them. For this, making use of the transformation function tm_map, I build a corpus in which I:

- Transform uppercase into lowercase. Tolower.
- Remove punctuation marks
- Remove numbers
- Remove extra white spaces
- Apply stop_words to cancel the most common words that are not relevant for the analysis

Unigram-bigram term frequency analysis. TF-IDF:

Once I have clean and prepared the corpus, I apply the DocumentTermMatrix function and obtain a matrix that reflects, for each restaurant, the words and its weight or frequency (number of times it is repeated in the corpus of that restaurant). As the matrix obtained has a very high level of sparseness, I apply a filter to select words that have a level of **sparseness not greater than 75%**. This leads me to consider **273 referent characteristics** to the comments of the **reviews**. Next, I show a display with the 10 most frequent and 10 least frequent words:

	name	count
1:	place	115093
2:	good	111612
3:	food	108954
4:	like	80757
5:	great	69592
6:	just	65703
7:	get	64505
8:	order	64338
9:	time	61340
10:	one	58842

Table 1: Top 10 most frequent words

	name	count
1:	plus	5218
2:	fact	5396
3:	clean	5551
4:	wont	5633
5:	either	5732
6:	wish	5938
7:	fine	5984
8:	theyr	6063
9:	yet	6128
10:	point	6156

Table 2: Least 10 frequent words

I apply TF-IDF to penalize the words that are very frequent in the conversations of the different restaurants.

I apply the same technique, DocumentTermMatrix, to obtain the characteristics of the different types of cuisine. In this case I don't filter for a certain sparseness, and I get **123 new features**. I analyze the 10 most frequent and the least.

	name	count
1:	'restaurants'	13164
2:	'american	2114
3:	'sandwiches',	1513
4:	'chinese',	1173
5:	(traditional)',	1124
6:	'pizza',	1094
7:	'japanese',	1010
8:	(new)',	979
9:	'mexican',	952
10:	food',	892

Table 3: Top 10 frequent cuisine types

	name	count
1:	'barbeque'	1
2:	'egyptian',	1
3:	'australian',	2
4:	'senegalese',	2
5:	'scottish',	2
6:	'colombian',	3
7:	'venezuelan',	3
8:	'comfort	4
9:	'puerto	4
10:	rican',	4

Table 4: Least 10 frequent cuisine types

We see that the type of cuisine that has a greater frequency is "restaurants." It's a label that almost all restaurants have. It is a feature that does not add any relevant value to the predictive model, so we eliminate it.

We group the characteristics that we have obtained from the reviews (term frequency), with the type of cuisine referents, and with those provided as additional features: zip code, number of reviews, average rating. Regarding these additional features, I eliminate outliers in the feature *number of reviews,* so that not distort the results of the predictive model.

Finally, our predictive model has **398 features**.

I apply the different predictive algorithms:

1.- Logistic regression: I configure it to perform 3 iterations

2.- Decision tree: I also configure it to perform 3 iterations and set the following parameters

- minsplit: lower = 10, upper = 50
- minbucket: lower = 5, upper = 50
- cp: lower = 0.001, upper = 0.2

3.- Random Forest: I set the following values to the configuration parameters

- ntree: lower = 50, upper = 500
- mtry: lower = 3, upper = 10
- nodesize: lower = 10, upper = 50

4.- Xgboost

- nrounds: lower=200,upper=600
- max_depth: lower=3,upper=20
- lambda: lower=0.55,upper=0.60
- eta: lower = 0.001, upper = 0.5
- subsample: lower = 0.10, upper = 0.80
- min_child_weight: lower=1,upper=5
- colsample_bytree: lower = 0.2,upper = 0.8

I repeat the analysis exercise from the beginning, addressing another strategy to identify relevant features for the predictive model. In this case, I analyze bigrams instead of individual words. For the bigrams, the level of sparseness is much higher than the ones in unigrams. **I eliminate the bigrams that have a sparseness level higher than 90%**, which means that I have **68 characteristics related to the reviews**. These are the most and least frequent.

	name	count
1:	go back	4733
2:	pretti good	3980
3:	come back	3848
4:	realli good	3675
5:	food good	3200
6:	first time	3181
7:	feel like	2989
8:	can get	2933
9:	great place	2905
10:	next time	2901

Table 5: Least 10 frequent bigrams

	name	count
1:	place like	1335
2:	want tri	1345
3:	look forward	1355
4:	best ive	1376
5:	tri place	1389
6:	felt like	1400
7:	definit go	1409
8:	will back	1412
9:	food servic	1423
10:	dont get	1424

Table 6: Least 10 frequent bigrams

With the 68 characteristics related to reviews, I build new predictive models by applying the algorithms indicated above. **My new predictive model has 193 features**.

Mutual Information analysis:

I apply the function of R ***rword2vec*** to identify in the corpus, which words are closest to words related to hygiene and cleaning issues. Once the model is prepared, I select the **50 words closest to the vector**: clean, dirty, cleaning, hygienic, toilet, quality. The result is:

	word	similarity to a
1	toilet	0.9271060
2	bathroom	0.8233061
3	restroom	0.7747261
4	stall	0.7198906
5	clean	0.6853536
6	flush	0.6794558
7	towel	0.6497674
8	bidet	0.5946246
9	unisex	0.5806470
10	sink	0.5648111
11	urin	0.5564384
12	women	0.5531101
13	cleanli	0.5437996
14	sanit	0.5430691
15	graffiti	0.5412159
16	filthi	0.5389890
17	lock	0.5271185
18	dirti	0.5241084
19	scrub	0.5227756
20	spotless	0.5183387
21	pee	0.5157260
22	tp	0.5154990
23	dispens	0.5018856
24	soap	0.4889017
25	trash	0.4878465
26	washroom	0.4868153
27	latch	0.4818862
28	men	0.4809327
29	tidi	0.4754696
30	filth	0.4639730
31	potti	0.4633801
32	grotti	0.4593866
33	garbag	0.4588215
34	wipe	0.4559606
35	clean"	0.4547656
36	carpet	0.4521063
37	facil	0.4474100
38	bleach	0.4442885
39	wash	0.4396207
40	lysol	0.4385527
41	disinfect	0.4355114
42	cobweb	0.4344056
43	damask	0.4324700
44	bin	0.4303426
45	gender	0.4253065
46	wellmaintain	0.4249123
47	bathroomth	0.4205323
48	countertop	0.4197791
49	mirror	0.4170810
50	recycl	0.4168150

Table 7: Top 50 words close to vector hygienic

I select these words as features related to reviews. Together with those related to the type of cuisine and the additional ones (zip code, number of reviews and average rating), I have a data set of **175 features** with which I build the different predictive models, applying the algorithms indicated at the beginning of this report.

In order to check the accuracy of the different predictive models that I have been modeling, and since I had no labels or method of checking the accuracy for the rest of the restaurants, I have considered 80% of the 546 restaurants with known results (label) for the training of the model, and the rest of 20% for the predition and measurement of precision. Specifically, I calculated parameter F1. The result is the one I show in the following table:

	Logistic Regression	*Decision Tree*	*Random Forest*	*Xgboost*
Words. TF-IDF	0,5454	0,4597	0,6725	0,5192

Bigram	0.6955	0,5333	0,6153	0,5591
Mutual Information	0,5625	0,5057	0,6061	0,5072

We have obtained the highest values for the Logistic Regression algorithm, followed by the Random Forest algorithm.

It makes sense, since they are algorithms that work best when most of the features we analyze are linear and not categorical, as is the case we are analyzing.

On the other hand, I was a little surprised that, as regards the strategy of Feature Engineering of Mutual Information (words related to cleaning issues), it has generally given less accurate results. I interpret this as that the issues of cleanliness and hygiene do not usually, except in very extreme cases, be a precise object of conversation, and we must look at other parameters of the conversations such as poor quality, etc.