HARVARD UNIVERSITY

CS281

Final Project Update: Classification for the Yelp Data Set Challenge

Nicolas Drizard & Virgile Audi

December 2, 2015

1 ABSTRACT DRAFT

Our objective is to improve the categorization system of the Yelp businesses using machine learning techniques. To achieve our goal, we will extract significant information out of the text reviews posted on Yelp using a semi-supervised Latent Dirichlet Allocation model. We will then combine these results with characteristic features of the business - such as localizations, price, customer type, check-in data - and predict the categories with two approaches: a classification with a Multinomial Logistic Regression and a clustering method with a K-Nearest Neighbors. We will provide a way to assess the quality of our new categories. On one hand, we will validate existing categories by holding out a set a venues, perform predictions and compare with the existing manually-entered categories. On the other hand, we will assess the performance of the LDA model using the perplexity metric.

2 NUMERICAL PROCESSING AND FIRST CLUSTERING EXPERIMENTS

2.1 NUMERICAL FEATURE EXTRACTION

This part focuses on two tables from the Yelp data: business and checkin. The first one stores information about the businesses (localizations, name, categories ...) and the second contains the average number of customers checkins for each hour of the week.

We joined and converted these two tables in order to have only numerical or binary features and applied different operations:

- Dropping irelevant features
- Identifying categorical features and converting them into orthogonal vectors of a N dimensional space, with $N =$ *number of categories*
- One attribute of each business stores different information in an unstructured way (each business does not have the same number of information stored by this attribute). It represents, for instance, the price, the ambience, if alcohol is offered... We needed to identify the most shared informations to avoid too many missing values

On top of these steps, we chosed to focus first on the restaurants (the yelp data set contains reviews about dentists, supermarkets, etc.). Also to build easily our first baselines we chosed not to handle the missing values and droped the business with not enough information.

We therefore reformatted and sampled the original data set to form a new one with 205 features relatives to the customers checkins and the inherent properties of the restaurants. We will test these features under supervised learning algorithms.

Some improvements still need to be addressed:

- Combining features and/or applying them polynomial or cosinus base function to increase their complexity
- Considering more attributes and the business where we don't have the checkin information while filling the missing values. This inference can be done through the average or median values over the entire data set, or through a nearest neighbors estimation.

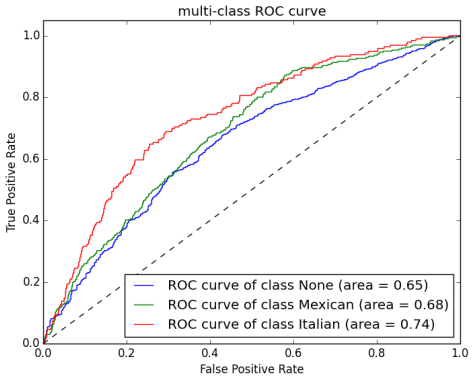
2.2 Supervised Learning

We applied a multiclass logistic regression from scikit learn as one of our baselines. The first question was to narrow down our targets. In the restaurants entries, there are still 261 different categories shared. As the text mining of the reviews seemed more complicated, we needed to consider the most diffenriating categories to have decent models. As a result, we focused on the nationality of the food to test among the restaurants.

With 3 classes: 1 without specific nationality and the two most represented nationalities, Mexican and Chinese, the multiclass logistic regression provided poor results on the test set (the data set was separated in the schema 80:20, train:test), only slightly better than predicting no nationality for each entry (dummy model). Some explanations could lie in that the features do not contain relevant information to differentiate the style and the restaurant set without missing values contains only 13000 entries with around 1000 positive for the two classes.

Precision score:

$$\rho_{dummy} = 0.671423029551$$
$$\rho_{logreg} = 0.726111608768$$



2.3 Next steps

Given the poor results of this part, we will focus on building a strong LDA for the next steps to model the categories of the restaurant. These new features could then be refined with the numerical features built.

3 Text preprocessing and LDA

3.1 Text preprocessing

The reviews came in the form of json file with each text review associated to a business id. Inspecting the data, we can find reviews of a line only and reviews much longer. Our motivation for this project is to classify the venues based on the text reviews, so the identity of the reviewers does not matter. We therefore merged the reviews by venues. We reduced the 1.6 million reviews to a corpus of 60785 larger texts, 1 per business. Note that we first had to clean the texts by removing capital letters, digits, etc...

As LDA uses the bag-of-words assumption, we only wanted to keep words that had a semantic interest. So we removed a list of stop words based on the most common words in the English language yielding a total vocabulary of over 400 thousands unique words. We then transformed the corpus into a sparse matrix for which each row corresponds to a document and column j tracks the count of the word indexed by j in the vocabulary. This step made our computers' ipython kernel crash multiple time but we finally obtained a sparse matrix of size 1 GB.

Following the choice in the numerical feature processing, we focused only on restaurants. Reprocessing everything yielded a corpus of about 18000 venues, and a total vocabulary of 200 000 words. We then persued to fit a baseline LDA model using the lda python package. But the computation ended up being way to expensive timewise (> 8 hours). We subsambled 500 restaurants, which reduced the size of the vocabulary to only ~40 000 words and ran the algorithm again. Completion time for the algorithm was about an hour for 50 topics. Some of the results are shown below:

```
In [19]: for i, topic_dist in enumerate(topic_word):
             topic_words = np.array(word_to_idx.keys())[np.argsort(topic_dist)][:-n_top_words:-1]
             print('Topic {}: {}'.format(i, ' '.join(topic_words)))

Topic 0: ensued pendant uninspired direkt emerald creations freee laptop nola
Topic 1: smell galette corking direkt pakoora reluctant scatter restaurateurs cacha
Topic 2: uninspired emerald feutr adrienne ceramics gl entomatadas efficient washing
Topic 3: ciscos direkt bubbled crannied chese ischitana crick litchees dwarfs
Topic 4: rita revamped waitressed incompetant heathen beggars tazhiki seaweed spiking
Topic 5: crannied mlk bullet trixies har hephaestus bothers moira onlined
Topic 6: marischino perused cucumber entomatadas traveler jackson parrot vetted boris
Topic 7: efficient dolci ciscos wrecking crick sunnyd therefore doggy ensued
Topic 8: direkt capresese licieux chese efficient hostesses reuben overpoweringly upcharging
Topic 9: austentatious alexander hedge efficient ensued gil bubbled leverage frog
Topic 10: dufferent otra thursday listen clt veryyy tasse artificial navigating
Topic 11: bearing chz rechnung escapes yury reuben solant hongkong loosening
Topic 12: pakoora yury boneless paru disasters ill vetted cacha gaudy
Topic 13: direkt cacha bubbled gunning tightenings rattling steap licieux solant
Topic 14: direkt food_mexican_restaurants mentaire chillies pakoora ci preps friggen amaaaazing
Topic 15: sg underfunded discarded nontheless choros pjhl unappetizing viennoiserie boris
Topic 16: cacha solant coherently fidgeting snatches buildings belle seaweed angering
Topic 17: whata crapes diavlo nuoc chapati reuben dolci authorized cutesy
Topic 18: rooting fishy coverage shortcake sashima rivers hootin looked dufferent
Topic 19: viennoiserie slowly nontheless bbls pram byeee choros morgan intimidated
Topic 20: cobblestone nostalgia portugaise steap neice mashed comprehension soysauce loco
Topic 21: whitemeat cutout italiano slef lines direkt cucumber francais aggravating
Topic 22: livers barockgeb direkt bbls amiable buildings coverage weapon entomatadas
Topic 23: positiven awekward maitake importantly invitant adrienne sensuous agreeing anges
Topic 24: morgan moderne entomatadas jitsu hoc extras divided listen diavlo
Topic 25: brucetta jackson boris gremlins jitsu entomatadas crack tightenings baies
Topic 26: alchemy boning puddle refrained pawn cassava celeriac wwife visible
Topic 27: ailie smoothest pois buritto footlong curbs sanwiches biter slumber
```

The topics obtained are very poor and there are numerous reasons for that. The first is that we didn't get the chance to cross-validate the number of topics queried. The second is that we then noticed that we had an issue with the spelling. If any of the TFs have suggestions on how to tackle this issue, we would greatly appreciate it. A third reason might lie in the fact that we are failing to distinguish what constitutes the reviewer's opinion (is it good restaurant? is it bad?) and what is this restaurant about (Italian? Japanese?). This third reason motivated the analysis of a customed LDA model that we will describe below. We haven't yet done any type of analysis but this is the model that we will now focus on and code up. This will alow us to use the regular LDA as a baseline to compare new topics.

3.2 The custom LDA model

The assumption that we will make for this model is the following: each review can be divided into two parts,

- words coming from an opinion topic that correspond to the rating of the review (in this particular case, we will use 5 topics for 5 stars),
- and words coming from any type of content topics.

For simplicity reason, we will also assume that the proportion of words from each part is fixed, for instance 30:70 or 40:60. Depending on the performance of this new method, we might decide to relax this last assumption and try to learn this parameter as well.

The new generative process for each review r will therefore be:

1. Generate the length using a Poisson distribution: $N_r \sim \text{Poisson}(\xi)$
2. Choose an opinion topic: $o_r \sim \text{Cat}(\beta)$, where $\beta \in \mathbb{R}^5$
3. Draw the per-document content topic distribution: $c_r \sim \text{Dir}(\alpha)$
4. For each word w_{ri}, generate $u \sim \text{Unif}(0,1)$:
   a) If $u <= p$ where p is the fixed proportion ruling the opinion/content separation then:
      - Pick a word $p(w_{ri}|o_r) \sim \text{Cat}(o_r)$
   b) If $u > p$ then:
      - Choose a latent content topic: $z_i \sim \text{Cat}(c_r)$
      - Choose a word $p(w_{ri}|z_n) \sim \text{Cat}(z_n)$

The mathematics behind shouldn't be too different from the derivations in the regular LDA model. We shall investigate this new model in the next few days.