HARVARD UNIVERSITY

CS281

Final Project Update: Classification for the Yelp Data Set Challenge

Nicolas Drizard & Virgile Audi

November 7, 2015

1 ABSTRACT DRAFT

2 NUMERICAL PROCESSING AND FIRST CLUSTERING EXPERIMENTS

3 TEXT PREPROCESSING AND LDA

3.1 TEXT PREPROCESSING

An important part of these first days of work was processing the text reviews on which we would perform LDA. The reviews came in the form of json file with information such as:

- the business id of the business reviewed,
- the reviewer's id,
- the number of stars given
- the text of the review itself in the form of a string
- and some extra information such as the class of review (fun,useful and others) based on Yelp users' votes.

We were first interested in extracting the business ids and the text reviews. The length of the texts varied a lot. 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 such as: "is, a, I, 'm, 's, and, can, be, etc...". This yield a total vocabulary of over 400 thousands unique words, which dazzled us at first, but we'll come back on that issue later. 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 140 GB.

The size of the document-term frequency (dtf) matrix being too big to work with, we decided to focus only on restaurant type of businesses (the yelp data set contains reviews about dentists, supermarkets, etc.). Reprocessing everything yielded a corpus of about 18000 venues, and a total vocabulary of 200 000 words. Reevaluating the dtf matrix was then much easier. We then persued to fit a baseline LDA model using the lda 1.0.2 python package. But the computation ended up being way to expensive timewise (we left the algorithm run for 8 hours and still didn't converge even though we enforced a maximum number of iterations of 1500!). In order to show some results for this update, we subsambled 500 restaurants and ran the algorithm again. Subsampling had for consequence to reduce the size of the vocabulary to only ~40 000 words. 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 a 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, that might result to be very problematic, with 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.