**Document classification**

**Dataset description:**

The dataset we use is Amazon Fine Foods reviews, which consists of 568,454 reviews of fine foods from 256,059 users left up to Oct. 2012.

Amazon fine food reviews dataset has the following attributes: ProductId, UserId , ProfileName, HelpfulnessNumerator (number of users who found the review helpful), HelpfulnessDenominator (number of users who indicated whether they found the review helpful) , Score (rating of the product), Time , Summary(brief summary of the review)  and Text(text of the review).

In our document classification problem, we use text information, Summary and Text as the corpus to train and test the LDA model. Score, whose rating between one and five, is the categorical variable in the document classification model. By applying LDA model, we can use a dense document-topic matrix derived by LDA model to replace a sparse co-occurrence matrix, thus serving a function of dimensionality reduction.

However the original dataset is too large thus too time-consuming for us to train the LDA model, so we randomly choose 1500 reviews from 1478 users evaluating 1318 products.

Dataset: <http://snap.stanford.edu/data/web-FineFoods.html>

Reference: J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. WWW, 2013.

**Choose appropriate number of topics:**

Here we use perplexity to evaluate the language model. Perplexity is a measurement of how well a model for prediction, ie. the inability(uncertainty or confusion) in understanding the text. In Blei ,the perplexity is algebraically equivalent to the inverse of average log-probability for a test set of documents, and is computed as



\[perplexity({{D}\_{test}})=\exp \left\{ -\frac{\sum{\log P({{w}\_{d}})}}{\sum{{{N}\_{d}}}} \right\}\]

A lower perplexity means a lower uncertainty when predicting the unseen documents, indicating a better performance for prediction.

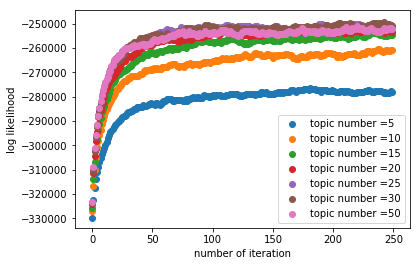
In this experiment, we separate the corpus into 90% for training and 10% for testing, and compute the perplexity under different number of topics. And then choose the topic number gives the lowest perplexity to do further analysis.

**Application on Comment text**

**Topics**

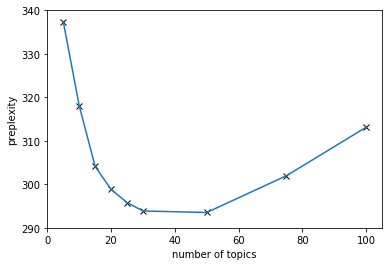
Choose topic number = 5, 10, 15, 20, 25, 30, 50, 75, 100.

Convergence test of Gibbs sampler



Figure【1】: loglikelihood under 0-250 iterations.

We plot the log likelihood under different number of iterations, from Figure[1], we can clearly find that after 250 iterations, the Gibbs sampler has already converge, and averagely speaking, for the document set of food review text, it will spend 2000 second(min = 1587s, max = 2796s) to train the LDA model and test it.

Then we compute the perplexity for the 10% test corpus under these topic numbers. 

Figure[2]: the relationship between topic number and perplexity.

Figure [2] shows that, 50 topics give the lowest perplexity, but since the perplexity of 30 topics and 50 doesn’t vary much (perplexity(30) = 293.868, perplexity(50) = 293.521) and it spends less time when building a LDA topic model under 30 topics (Time(30) = 1990.05s ,time(50) = 2212.37s), we choose 30 as the number of topics.

In each topic, we get the topic-vocabulary matrix represent the probability of the word given the topic. And we choose the ten largest probabilities of the words as the representative words of the given topic. The detailed result of 30 topics and their representative words is in the Appendix. Here we show some interesting topics and their representative words, and according to these representative words, we do manual annotation on the topics. Some of the topics are about the serves, like after-sale and delivery service in topic 1 and topic 18, some are about the category of products, like energy drink and coffee, and some are about the quality and health, etc.

|  |  |  |
| --- | --- | --- |
| Topic# | Representative words | Manual Aspect |
| Topic 1 | company, help, thought, please, stuff,  change, huge, issue | After-sale |
| Topic 3 | drink, taste, ad, juicy, diet, want, energy,  vitamin, like | Energy drink |
| Topic 8 | price, review, bag, great, brand, quality,  new, know, deal | Price & quality |
| Topic 11 | sweet, sugar, high, calories, fat, serve,  low, protein, healthy | Sugar, calorie & fat |
| Topic 18 | order, product, amazon, purchase, ship, item,  receive, arrive, seller | Delivery |
| Topic 27 | coffee, cup, pod, brew, roast, bold, strong,  machine, Keurig | Coffee |

Table [1]: Part of the topics, representative words and manual notation.

下面这部分可以放在appendix里面

=========================== topic number = 30 =============================

Topic 0: tea | green | flavor | spice | bag | make | chai | leav | tast

Topic 1: compani | help | thought | pleas | stuff | chang | huge | issu

Topic 2: food | cat | eat | organ | year | babi | month | old | start

Topic 3: drink | tast | ad | juic | diet | want | energi | vitamin | like

Topic 4: dog | treat | love | recommend | chicken | anyth | feed | dri | quick

Topic 5: tri | great | realli | bit | best | favorit | way | happi | nice

Topic 6: product | ingredi | contain | list | syrup | natur | label | corn | addit

Topic 7: good | better | star | pretti | bit | think | brand | someth | decent

Topic 8: price | review | bag | great | brand | qualiti | new | know | deal

Topic 9: box | packag | howev | open | pack | purchas | actual | sure | cost

Topic 10: enjoy | time | differ | fresh | worth | end | good | lot | notic

Topic 11: sweet | sugar | high | calori | fat | serv | low | protein | healthi

Topic 12: love | littl | peanut | butter | snack | think | good | eat | kind

Topic 13: bar | size | think | chew | eat | small | piec | half | larg

Topic 14: chocol | smell | like | someth | expect | nice | milk | bad | cream

Topic 15: hot | sauc | chees | spici | noodl | far | best | origin

Topic 16: buy | bought | store | local | groceri | free | thought | gluten | cheaper

Topic 17: flavor | tri | disappoint | fan | bad | good | realli | strong | hint

Topic 18: order | product | amazon | purchas | ship | item | receiv | arriv | seller

Topic 19: like | look | got | thing | real | right | anoth | make | way

Topic 20: water | bottl | tast | want | recommend | acid | long | bodi

Topic 21: tast | like | better | quit | vanilla | milk | strong | sort | opinion

Topic 22: time | day | everi | need | realli | minut | noth | jar | wonder

Topic 23: cooki | amazon | product | com | ounc | pack | www | gp

Topic 24: bag | oz | howev | know | come | pound | alway | read

Topic 25: tast | like | chip | salt | popcorn | sure | definit | salti | pop

Topic 26: use | work | feel | wast | hair | brand | good | condition

Topic 27: coffe | cup | pod | brew | roast | bold | strong | machin | keurig

Topic 28: appl | candi | tri | hope | help | want | ok | dri | stuff

Topic 29: make | mix | littl | add | use | easi | coconut | rice | cook

=========== End =========== (这部分如果要做成上图表格的话也可以)

**Auto-tagging**

According to the document-topic matrix and topic-vocabulary matrix, we can do auto-tag for a given text. From a Bayesian paradigm, we can compute the probability of each words given the certain document.



$P(w|D)\propto P(w|z)P(z|D)$

, where w is the word, D is the given document and z is the topic.

Choose the top 10 largest conditional probability as the tag of the given text D.

**Sentiment classification**

We can infer the sentiment behind the food review, and these sentiments, whether satisfaction or discontent, can be somehow reflected on the score rating – from 1 to 5. Intuitively, when we score 1 or 2, we are unsatisfied, conversely, when rating 4 or 5, we are pleased with the service or product. Here, we divide the sentiment into **two categories**, and **three categories.**

**Feature vector**: In original classification using word feature, we use 0-1 co-occurrence vector of a certain text as a feature vector, whose dimension is more than 2800. In a modified classification applying LDA model, we use the document-topic vector as a feature vector with a reduced dimension = 30 (as the analysis above, when topic number = 30, it gives the lowest model perplexity).

Given that in the original word feature model, the length of feature vector is more than 2800, which is larger than sample size 1500, the classification model we use is **logistic regression classifier.**

**Binary classification – positive (score 3, 4 or 5) and negative (score 1 or 2)**

Denote negative attitude as 0 and positive as 1,

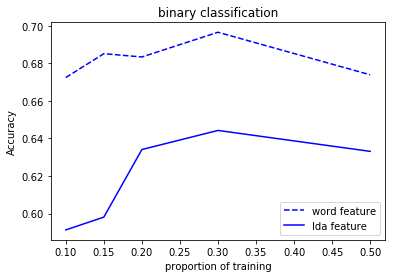


\[\hat{P}=\frac{1}{1+{{e}^{-\hat{\mu }}}},\text{where }\hat{\mu }=\hat{\theta }x\]

, and x is the feature vector. The decision boundary is {x: P = 0.5}

Then we compute the accuracy of the classification under different proportion of the test set. Accuracy is defined as the proportion of the number of correct classification.

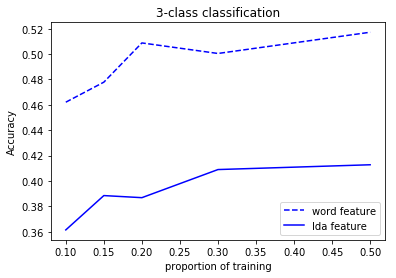
The reduction in dimension certainly results in accelerating the LR classification. But when dealing with accuracy, LDA model fail to classification well.



Figure[3]. Classification results on binary classification.

**Three-class classification – positive (score 4 or 5), neutral (score 3) and negative (score 1 or 2).**

Sometimes, score 3 is hard to define under 1-5 rating system, and in this case, we regard score 3 as a separate group – neutral, thus making the classification a multiclass one. Traditionally, logistic regression is used in binary, and fortunately, we can still use one-vs-all logistic regression classifiers, training a single classifier per class, with the samples of that class as positive samples and all other samples as negatives (ie. decision tree + logistic regression).



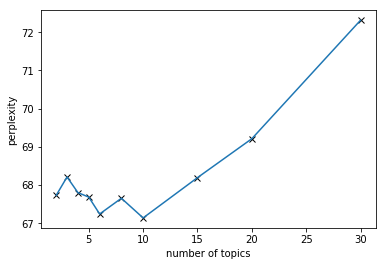
Figure[4]. Classification results on three-class classification.

From Figure[4], disappointedly, the LDA model also doesn’t well in multi-classification problem.

We believe the major reason is that the word feature contains all the information of the document, but LDA feature, after all, is a compressed one. Besides, we guess another reason is partly due to the attribute we use – Text, maybe text is a little bit long for LDA model to do classification, also running Gibbs LDA model on the review text too time-consuming. So, we decide to use shorter comment – Summary to do classification.

**Application on Comment summary**

Just like the steps in the application on comment text, we first compute perplexity and choose the appropriate topic number, 10, the results are following:



Figure[5].

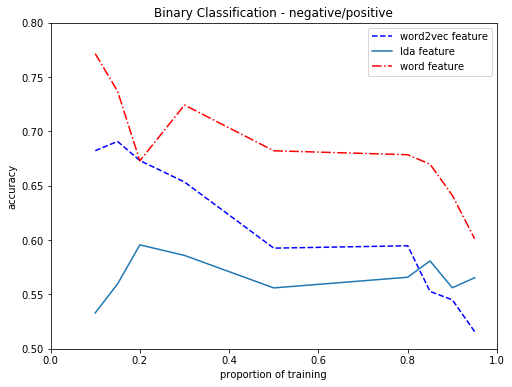
|  |  |
| --- | --- |
| Topic# | Representative words |
| Topic 0 | dark, flavor, smell, excite, ginger, green, like, amaze |
| Topic 1 | ok, like, make, taste, love, price, disappoint, food |
| Topic 2 | tasty, product, best, quantity, good, perfect, buy, yuck, size |
| Topic 3 | ice, strange, strong, salty, excel, candy, wonder, juicy |
| Topic 4 | buy, disappoint, weak, aw, clean, delicious, good, ginger |
| Topic 5 | great, taste, expect, butter, water, value, food, ok |
| Topic 6 | tea, product, dog, drink, green, little, juicy, candy |
| Topic 7 | cat, coffee, love, quality, cup, chocolate, great, tea |
| Topic 8 | price, food, tasty, taste, clean, bold, like, longer |
| Topic 9 | good, delicious, chocolate, eh, tasty, mustard, disappoint, ginger |

Table[2].

Besides the original word feature, we also consider a new method dealing with the feature vector - **Word2vec**. Word2vec is also a method to reduce the dimension of the feature vector for classification, its main idea is to estimate the distance between the word and its context words in a document. To simplify, we just directly use word2vec in *genism* package.

Classifier: Stochastic Gradient Descent (**SGD**), commonly used when the feature vector is sparse, is also a good classifier in NLP problems.

**Binary classification:**

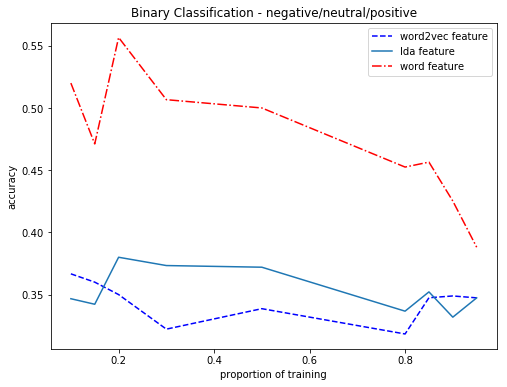


Figure[6].

From Figure[6], the original word feature is still the best one used in classification. However, the word feature is time consuming in the large-scale problems. Comparing word2vec feature and LDA feature, word2vec feature does pretty well when there isn’t much training data, and LDA feature is good after a large amount of trainings.

3-class classification:

From figure[7], we find that the original word feature works much better than LDA and word2vec feature, and gives a super good performance in 3-class classification. Unlike the binary classification, LDA model works better than the word2vec feature with the low proportion of the training data. Nevertheless, whether LDA or word2vec, both are mediocre when dealing with multi-classification.



Figure[7].

**The possible reason that LDA model does not work well.**

From the results and analysis above, no matter the long reviews or the short summaries, disappointedly, unlike the performance in Blei, the LDA classification is not as good as original word feature, and this is partly because of the dataset we use. We simply choose 1500 comments from the whole dataset, which makes the data somewhat “impropriate” for building model. For example, most of the 1318 products have only one review, thus making the topics untypical. Another problem is that the distribution of Score is not the same as the original dataset, which would also influence the performance of the LDA model.