Question

3.2.4

POS tagging is an essential step in understanding the syntactic structure of the sentence. The nltk library offers a lot of different kinds of POS taggers. We reply on the Brown Corpus as the training data to train a tagger.

The simplest tagger is to assume that all the words in a sentence follow the same POS tag which occurs most frequently in the training corpus. However, this tagger is not accurate as words have different kinds of POS tags. In our code, we name this tagger as the default tagger.

A better tagger would be rule-based and assigns POS tags to words based on its stem and suffixes. This tagger works better than the default tagger, however, it still assigns wrong POS tags sometimes as there are irregular words which do not follow the defined rules exactly.

By observation, we can see that there exists a tag which occurs most frequently, but the most frequent work may not has the most frequent tag. Thus, we can sample the most frequent 100 words and use their POS tags to train a unigram tagger as the baseline tagger. The weakness of this tagger is that it fails to assign a tag to an unseen word.

To further improve the tagging performance, we can adopt the N-gram method. We have created Uni-gram, Bi-gram and Tri-gram taggers with the Brown Corpus. These taggers have the option to use backoff. If they use backoff, a higher level tagger will resort to the results produced by the lower tagger if the higher-level tagger cannot be used.

Lastly, we resort to the current deep learning techniques to build a multi-perceptron tagger. The perceptron tagger learns the mapping from the words to POS tags in a high dimensional space.

We have created a class called POSTagger and to randomly tag 5 sentences, we can run the tag\_random\_sentences function.

A screenshot of a cell phone

Description automatically generated

As there is insufficient space to accommodate the tagging results produced by all taggers, only the shortest sentence in the results of the best performing perceptron tagger is shown here. You may refer to the tagger\_result.csv for all the results produced by all taggers.

('If', 'ADP'),('I', 'PRON'),('could', 'VERB'),('give', 'VERB'),('this', 'DET'),('place', 'NOUN'),('zero', 'NUM'),('stars', 'NOUN'),('I', 'PRON'),('would', 'VERB'),('.', '.'),('This', 'DET'),('place', 'NOUN'),('has', 'VERB'),('got', 'VERB'),('to', 'PRT'),('be', 'VERB'),('the', 'DET'),('most', 'ADV'),('disgusting', 'ADJ'),('Mexican', 'ADJ'),('place', 'NOUN'),('I', 'PRON'),('have', 'VERB'),('ever', 'ADV'),('ate', 'VERB'),('at', 'ADP'),('!', '.'),('!', '.'),('!', '.'),('!', '.'),('Their', 'PRON'),('meat', 'NOUN'),('was', 'VERB'),('overcooked', 'VERB'),('and', 'CONJ'),('flavorless', 'ADJ'),('.', '.'),('Salsa', 'NOUN'),('had', 'VERB'),('no', 'DET'),('flavor', 'NOUN'),('at', 'ADP'),('all', 'DET'),('!', '.'),('!', '.'),('Even', 'ADV'),('their', 'PRON'),('tortillas', 'NOUN'),('were', 'VERB'),('bad', 'ADJ'),('!', '.'),('!', '.'),('So', 'ADV'),('please', 'ADJ'),('do', 'VERB'),("n't", 'ADV'),('listen', 'VERB'),('to', 'PRT'),('want', 'VERB'),('anybody', 'NOUN'),('else', 'ADV'),('has', 'VERB'),('to', 'PRT'),('say', 'VERB'),('about', 'ADP'),('this', 'DET'),('place', 'NOUN'),('.', '.'),('It', 'PRON'),('should', 'VERB'),('be', 'VERB'),('illegal', 'ADJ'),('to', 'PRT'),('sell', 'VERB'),('food', 'NOUN'),('this', 'DET'),('bad', 'ADJ'),('!', '.'),('!', '.'),('!', '.')

3.2.5

The first task is to find the top-10 most frequently used adjectives for each rating star and the second task is to find the top-10 most indicative adjectives for each rating star. The scripts of both tasks are encapsulated into a class called AdjExtractor in the most\_freq\_adj.py. As these tasks require rating-star-specific analysis, the reviews are first segmented into 5 csv files based on the rating star using the group\_reviews\_by\_rating function.

A screenshot of a cell phone

Description automatically generated

For the first task, we need to count the frequency of each adjective in each rating star and then get the adjectives which have the top-10 frequency. The results are written into the most\_freq\_adj.csv by running the extract\_top\_ten\_most\_freq\_adj funtion of the AdjExtractor instance.

A screenshot of a cell phone

Description automatically generated

The results are shown below. 1 means the most frequent and 10 means the 10th most frequent.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 star | 2 star | 3 star | 4 star | 5 star |
| 1 | good | good | good | good | great |
| 2 | other | other | other | great | good |
| 3 | bad | great | great | nice | friendly |
| 4 | first | bad | nice | other | delicious |
| 5 | new | nice | little | little | nice |
| 6 | last | first | small | delicious | other |
| 7 | horrible | much | bad | fresh | amazing |
| 8 | few | little | few | friendly | fresh |
| 9 | same | few | decent | small | first |
| 10 | next | small | hot | first | new |

It can be observed that as the rating star goes down, the occurances of negative adjectives such as “bad” and “horrible” increases. As the rating star goes up, the occurances of positive adjectives such as “delicious”, “amazing”, “friendly” and “decent” increases. This demonstrates that low rating comments tend to have more negative adjectives while high rating comments tend to have more positive adjectives. In addition, we can also see a lot of words which are not reflective of the consumers’ opinions or sentiment. Words like “other”, “last”, “same” and “next” are merely neutral and does not indicate any emotion. Interestingly, we can notice that the most frequent adjective for all rating stars is all positive, being either “good” or “great”. This is perhaps because we only consider a single word adjective without taking the negation words such as “not” into consideration. We suspect that the in low rating comments, there is a high change of “not good”, thus, the frequency of the word “good” is high.

For the task two, we are required to find the top-10 most indicative adjectives for each rating star. For example, the indicativeness of an adjective in rating star 1 can be computed using the following equation.

A picture containing object

Description automatically generated

Where P(w | R1) is the probability of observing word w in all reviews with rating star 1 and P(w) is the probability of observing the word w in all reviews.

The results can be computed and written into the most\_indicative\_adj.csv file by running the extract\_top\_ten\_most\_indicative\_adj function.

A screenshot of a cell phone

Description automatically generated

The resutls are shown as below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 star | 2 star | 3 star | 4 star | 5 star |
| 1 | horrible | bad | good | good | great |
| 2 | terrible | ok | decent | great | friendly |
| 3 | rude | disappointed | ok | little | amazing |
| 4 | bad | same | other | delicious | delicious |
| 5 | poor | dry | small | nice | awesome |
| 6 | awful | chinese | average | tasty | happy |
| 7 | last | much | little | small | excellent |
| 8 | disaapointed | decent | overall | fresh | wonderful |
| 9 | unprofessional | disappointing | nice | only | helpful |
| 10 | same | second | bad | hot | professional |

Compared to the results in the first task, the results in the second task shows a stronger correlation between the star rating and the consumers’ sentiment. The number of negative adjectives is larger than that in the first task for low rating comments and the number of positive adjectives is larger for high rating comments as well. These adjectives are most indicative of the consumers’ sentiment towards the resturants.