CSE 398/498 Text Mining

Project 1

Sihong Xie

October 6, 2016

1. Project Description

1.1 Main goals

In this project, you (in a single person team) are asked to implement a classifier and an information extractor for sentiment analysis. Given reviews of many products in the category of musical instruments from Amazon (see the dataset description), accomplish the following tasks:

1. Predict the rating from the review texts.
2. Extract attributes and opinions from the reviews, with evaluations.
3. Pair opinion words with attribute words, such that the opinion is describing the attribute, with evaluations.

1.2 Details

1.2.1 Task 1

First you need to pre-process the text data, including tokenization and normalization (you're asked to think about which normalization would be suitable). After you have built a vocabulary of size d, represent each review as a d-dimensional vector (recall the vector space model). Use TF-IDF weighting introduced in class or the IIR book. Then randomly partition the vectors into two subsets, namely, the training (80%) and test (20%) datasets. Also discretize the ratings as follows: ratings of 4 and 5 are positive ratings, and the review will be in class 1; other ratings from 1 to 3 are considered negatives, and the reivew belongs to class 0. Train two classifiers (naive Bayes and Rocchio) using the training portion and test its performance on the test portion. Report the running time of training and testing, and the accuracy, precision and recall.

1.2.2 Task 2

Task 2 involves POS tagging [1]. Use the Stanford POS tagger to tag each sentence, find out the words tagged as NN (singular nouns) and NNS (plural nouns), and words that are tagged as JJ (adjectives), JJR (comparative adjectives), and JJS (superlative adjectives). The nouns will become your candidate attributes and the adjectives the candidate opinions. You are responsible for thinking ways to get rid of irrelevant nouns and adjectives (this step is considered optional and has bonus scores). For example, the following sentence

"The portfolio is fine except for the fact that the last movement of sonata #6 is missing ."

will be tagged as the following sequence of <word, tag> pairs:

[1] covered in Lecture 12, but you don't have to wait until then, as the example codes will give you what you need for this project

[('The', 'DT'), ('portfolio', 'NN'), ('is', 'VBZ'), ('fine', 'JJ'), ('except', 'IN'), ('for', 'IN'), ('the', 'DT'), ('fact', 'NN'), ('that', 'IN'), ('the', 'DT'), ('last', 'JJ'), ('movement', 'NN'), ('of', 'IN'), ('sonata', 'NN'), ('#6', 'CD'), ('is', 'VBZ'), ('missing', 'VBG'), ('.', '.')]

In this example, "portfolio" and "fact" will both be extracted as attributes, but "fact" is not a valid attribute. Similarly, "fine" and "last" are candidate opinions, but only "fine" is a valid opinion word. Report the top 50 attributes across all reviews with highest frequencies. See the Deliverables section for details.

1.2.3 Task 3

For Task 3, implement a naive baseline that simply regards all (attribute, opinioin) paris appearing in the same sentence as a valid pairs. In the above example, ("portfolio", "fine") is a valid pair, but ("fact", "last") is not. Report the pairs that have one of the top 50 attributes. If there are less than 50 such pairs, use the attributes following the top 50 as candidates. You can choose to improve these results, and you may earn bonus if the improvement is reasonably good. See the Deliverables section for details.

Evaluations of the above extracted results (task 2 and 3) are critical. Manually check whether the extracted attributes are indeed aspects of the corresponding product, and the opinion word is indeed used to describe that attribute for each pair. To do this, you need to keep and investigate the sentences from which you extract the pairs to make sense of the extracted information. See the Deliverables section for reporting your evaluation results.

1.3 Dataset

Please use the link [2] to download the dataset (about 2.3M). You actually don't have to worry about the format of the dataset, as I've included a function in the supporting codes to extract the text and rating for each review. Your program should rely on this function and focus on text processing.

1.4 Softwares

Python (3.x) and Java (1.8) are required for this project. NLTK (text pre-processing), scipy and numpy (matrix and linear algebra), scikit-learn (classifiers) are standard packages included in Anaconda. You shall be able to use them without trouble.

The POS tagger is from Stanford NLP group. It is basically a pretrained classification model written in Java, and you will use the model to predict the part-of-speech of the words in the sentences. There are two ways to use the software: you can either invoke Java interpreter on the command line (see the webpage for the POS tagger) or use Python to call the Java program. For you to follow the second way, download the necessary packages and supporting codes on the course website, and set up the path in the Python codes to reflect the location of the downloaded packages on your local machine. For details, please read the comments in the notebook.

[2] http://www.cse.lehigh.edu/~sxie/teaching/data/reviews_Musical_Instruments_5.json.gz

2. Deliverables

Your submission contains two parts:

- A report of details of how you design your programs. This part is for me to understand your codes and convince me that your codes is doing something reasonable and meaningful to accomplish the above goals. Also narratively include in the report your findings when you evaluate your results. For example, if you find out that the most frequent nouns are actually invalid aspects, then you may want to share that with me.

- The codes in a zip package. You can use any programming language (but I will be more helpful when you choose Python). Experimental results shall be stored in three files:

    1. A PDF file containing the results (running time, performance, etc.) of the first task shall be reported in a figure or table. No other formats will be accepted. You may use LaTeX or Word (converted to PDF).
    2. For task 2, the extracted attribute words, their frequences and your evaluation shall be reported in the file named "p1_t1.csv". In the text file, an attribute word, its frequency and evaluation shall occupy a single line. The format of a line looks like

       portfolio,105,1,"<a sentence where portfolio appears>"

       where **porfolio** is the extracted noun, and 105 is its frequency. Following these two, if you think that the attribute word is mistakenly extracted (such as "fact"), put down 0 as its label, and if attribute word does denote a proper aspect of a product (like "portfolio") then put down 1 as its label. Following the label is a sentence of your choice that supports you evaluation of the attribute.
    3. The format of the results of the third tasks is similar to that of Task 2, but you will report the (attribute, opinion) pairs, your evaluation and a supporting sentence where the pair appear. The file name is "p1_t2.csv", where a line is desinated for a single discovered pair:

       (portfolio,fine),1,"<a sentence where the pair appears>"

       The first field is the pair in parentheses, with a comma separating the attribute and the opinion words. If the pair is regarded as valid (the opinion word does describe that attribute), then put down 1, otherwise 0 in the second field. The third field is a supporting sentence enclosed in quotations. No need to report the frequencies of the pairs. I have included a function in the Python notebook for output the results, you may use that function for output purpose. If you choose other languages, make sure the output follows the above format.

3. Grading

The total score is 100 points. The clearness and readability of the report (the first part of the deliverables) will account for 30 points: the less time I have to spend to understand your codes, the more points you can get. The remaining 70 points go to the codes and the results, which are judged based on the correctness and usefulness. Specifically, the codes shall be able to run smoonthly and produce the required results (correctness). Furthermore, the more valid attributes and <attribute, opinion> pairs your codes can find (I will be the final judge here), the better (usefulness). A bonus of 30 points is given to the two optional improvements for task 2 and 3 (15 points each).