Natural Language Processing – Assignment #2

Assignment description

This assignment includes two tasks – (1) questions about the Viterbi decoding algorithm, and (2) text classification task. Please write your concise answers to task #1 in a document.

Task #1: the Viterbi algorithm for sequence decoding

1. Does Viterbi guarantee to find the optimal solution? That is, does it find a sequence with the highest probability, given a sequence of observations (a sentence)? Explain shortly (in your own words), referring to structures and the algorithm we saw in the class.

2. During forward pass, the algorithm assigns each cell in the matrix C a value according to the following computation:

$$\text{(a)} \quad C(i, j) = \max_k C(k, j\text{-}1) * A(k, i) * B(i, \text{ind}(w_j))$$

   Dropping the first factor in this computation will result in the below assignment:

$$\text{(b)} \quad C(i, j) = \max_k A(k, i) * B(i, \text{ind}(w_j))$$

   Give a small example for transition and emission matrices, where the alternative (b) is insufficient for computation of the most probable sequence, but the original alternative (a) would get it right. Example with two tags and two words should suffice for this purpose.

Task #2: text classification – Amazon products reviews

This task deals with text classification, one of the most common supervised tasks on text. A set of reviews in three diverse domains is attached to this assignment, and you goal is to predict a review's score (rating) based on its text. The ratings span values 1-5, meaning that is a 5-class classification.

Each review has the following format:

{"overall": 4.0, "verified": true, "reviewTime": "09 7, 2015", "reviewerID": "A2HDTDZZK1V4KD", "asin": "B0018CLO1I", "reviewerName": "Rachel E. Battaglia", "reviewText": "This is a great litter box! The door rarely gets stuck, it's easy to clean, and my cat likes it. I took one star off because the large is really small. My cat is 6 pounds and this isn't quite big enough for her. I'm ordering this same brand in a large. Online price is much cheaper than pets stores or other stores!", "summary": "Great Box, Get XL size!", "unixReviewTime": 1441584000}

where "overall" refers to the rating (the "label" you learn to predict), "reviewText" to the body of the review, and "summary" to its summary.

Reviews are split into train (with 2000 reviews per class), and test (with 400 reviews per class) – you train a classifier on train data, evaluate and report the results on test data. We don't have a validation set in this case since you are likely to work with classifiers' default parameters, i.e., no tuning is required.

You are given two files: `config.json` and `main.py`, where `config.json` contains train and test files location. You can change these locations in `config.json`, but not the code parsing it in the main. Your task is to implement the function `classify()`:

```
def classify(train_data, test_data):
      …

      return test_results
```

Note that you return a dictionary with the F1 metric per each class, as well as the overall accuracy. The ultimate goal is to obtain the highest possible accuracy on the classification task.

Implementation details and comments:

1. Use out-of-the-box classifiers provided by the python's scikit-learn (https://scikit-learn.org). The package provides means for converting text to vector representation using the classes `CountVectorizer` and `TfidfVectorizer`; a full pipeline example can be found in this tutorial – you can use its parts, understanding how things work and exploring the functions' parameters.

2. Use words, and possibly word n-grams as features; you can use additional features, but make sure your code complies with the runtime requirements.

3. Reduce your vocabulary to top-K most frequent words (or word n-grams) in the entire train corpus, where K is below 1000. Adding more words to the vocabulary will increase your processing time, and only marginally (if at all) contribute to the final accuracy.

4. Print your confusion matrix; which classes share the highest confusion? Why? Write your short interpretation of the confusion matrix in the report you're submitting.

5. Python's scikit-learn allows extraction of K features (words or word n-grams in our case) that have the highest discriminative power, i.e., are the best for the classification task at hand. One such function is `SelectKBest` provided by scikit-learn. Use this function to extract 15 features most effective for classification and report them at the document you're submitting.

6. Perform cross-domain classification: train on one domain (e.g., sports training data) and test on another (e.g., pets test data). How do your results compare to in-domain classification? Interpret your results shortly in the document.

7. Make sure your code runtime doesn't exceed 5 minutes (invocation → results are printed).

Submission

Submit a single zip file – assignment2_ xxxxxxxxx_xxxxxxxxx.zip , where “xxxxxxxxx” stands for a student id. Please specify two student ids (your and your partner’s). It should include two files:

1. A document with your answers for task #1, and points 5, 6 and 7 in task #2.
2. Your implementation for task #2 -- `main.py`.

Grading criteria include: correctness (the major part), code design, readability and documentation.

Good Luck!