Homework 1: Supervised Text Classification (100 points)

Kathleen McKeown, Fall 2021
COMS W4705: Natural Language Processing

Due 10/6/21 at 11:59pm ET

Please post all clarification questions about this homework on the class EdStem under the “hw1” folder. You may post your question privately to the instructors if you wish.

Overview

The goal of Homework 1 is to gain experience with supervised text classification using hand-crafted features. To do this, you will be 1) implementing classifiers for predicting debate winners (§1) and 2) analyzing features and n-gram smoothing methods (§2).

The classifiers you implement will be trained on a dataset of debates and observers’ opinions about the debates [1]. Your goal is to predict whether the PRO or CON debater was more successful. The debater that achieves the higher number of total points assigned by voters, across several dimensions, is declared the winner. You will be provided with two data files: for training and for development. Skeleton code and the data files are provided to you on CourseWorks.

We will run your model(s) on a hidden test set and **5** points of you grade will be based on how your model(s) performs on this test set.

- You will get one point if your model gets higher than 65% accuracy
- You will get an additional two points if model gets higher than 70% accuracy
- You will get an additional two points for if your model get higher than 72% accuracy

The top **5** submission in the class will also get bonus points.

1 Programming portion (50 points)

1.1 Tasks

In this assignment, you will implement several models using different feature sets and evaluate each model's accuracy. You will be implementing a classifier to predict the winner of a debate, given the debate text along with information about the debaters and the audience.

The goal of this assignment to understand what factors are important in determining the winner of a debate. In particular, you are given a set of debates where the debaters are from opposing religious ideologies (one is Christian and the other is Atheist), and you would like to understand what sets of features are useful for predicting the outcome of debates on **religious topics**, vs. **debates on all other topics**. Your task is to create an experimental design to understand the impact of these factors for these two groups of debate categories. *Hint: You should look closely at the training and validation data you are provided, and think about how you would partition the data to help you answer this research question. Note that you may need two separate models for each of the settings to measure the impact of debate topic*

Input Linguistic features derived from the debate text, along with features derived from user information as well as lexicons.
Output a label $y \in \{Pro, Con\}$, indicating whether the Pro or Con debater was more successful (i.e., Pro is more successful if they received more points from the audience voting).

The set of models you must implement are (see §1.1.1 for feature details):

1. *Ngram* model: this model should use **only** word ngrams and for each word ngram should represent its tf-idf score.
2. *Ngram+Lex* model: this model should use
   - word ngrams
   - lexicon-based features: implement lexicon based features for a lexicon of your choice.
3. *Ngram+Lex+Ling* model: this model should use
   - word ngrams

- lexicon based features (as in *Ngram+Lex model*)
- **2** Linguistic features from §1.1.1.3

4. *Ngram+Lex+Ling+User* model
    - word ngrams
    - lexicon based features (as in *Ngram+Lex model*)
    - **2** Linguistic features from §1.1.1.3
    - **2** User features from §1.1.1.4

For each model above, you should implement a logistic classifier that gives the highest **accuracy** on the development set and include only this in your code. Note that you may need two separate models for each of the settings above to measure the impact of debate topic. If this model has tunable parameters, you should tune them to perform well without overfitting. You may tune parameters you find in the scikit-learn documentation. In order to achieve good performance, you may experiment with choosing the k best features (e.g., k best n-grams), with different values for k.

Your submission must include all deliverables specified in §3.1 and must run in our environment.

1.1.1 Feature Details

Each of your models will use features detailed here:

1. **Word Ngrams** ($n = 1, 2, 3$). The value for a word ngram will be its tf-idf score (not its frequency). For introduction to tf-idf, you can refer to section 6.5 of the Jurafsky and Martin book [1].

2. **Lexicon-based features**
    (a) Lexicons
        i. Connotation lexicon [2] [2]
        ii. NRC-VAD Lexicon [3] [3]
    (b) How you extract features is part of the design decision that you need to make. One simple example for lexical features could be counting how many words in each debaters language appear in the corresponding lexicon.

[1] https://web.stanford.edu/ jurafsky/slp3/6.pdf
[2] https://www3.cs.stonybrook.edu/ ychoi/connotation/
[3] https://saifmohammad.com/WebPages/nrc-vad.html

3. **Other linguistic features**
    (a) Length
    (b) Reference to the opponent
    (c) Politeness words
    (d) Swear words
    (e) Personal pronouns
    (f) Modal verbs
    (g) Misspellings
    (h) Links to outside websites
    (i) Numbers
    (j) Exclamation points
    (k) Questions
4. **User features** This dataset contains a wealth of information about users such as their demographic information, their opinions on big issues, their social network, and their participation. As prior work [1] has shown, user information can be very useful in predicting the outcome of debates. Here you will need to think about how this user information can be used improve prediction accuracy, implement features that utilize this information and incorporate them into your model.

1.2 Resources

1.2.1 Skeleton Code

You will be provided with a file `hw1.py` that takes 4 command line arguments. Do not modify the command line argument processing or we may not be able to run your code. For mode details see §3.1.

1.2.2 Allowed external resources

For the assignment you are allowed to use scikit-learn implementations of models, feature selection algorithms, and metrics. You may also use standard python packages such as `pandas` and `numpy`. Do not use any other publicly available code (e.g., github repos). If you are unsure about using a package, please ask. Apart from these exceptions, you must write all code yourself. Please refer to the Academic Integrity policy if you have questions in this regard.

As an example, You can look at the text classification example in scikit-learn called “document_classification_20newsgroups.py”.

1.2.3 Data Format

The debates data (train and validation) is provided in json lines format [4]. The keys in each json record are as follows:

- "id": The id of the debate.
- "category": The category of the debate (ex: religion, politics, etc.).
- "title": The proposition of the debate.
- "rounds": This is a list of rounds in the debate. Each round is a list of dictionaries where each dictionary has a key called "text", that contains the text of the argument put forth by the given debater, and a key called "side" that specifies whether that argument was put forth by the Pro or Con debater.
- "date": The date the debate took place.
- "pro_debater": The name of the debater that is arguing for the proposition.
- "con_debater": The name of the debater that is arguing against the proposition.
- "voters": A list containing the names of the voter who voted in the debate.
- "winner": The winner of the debate as determined by the voters (i.e. Pro or Con).

2 Written portion (50 points)

Please write all answers in a single file that you will submit as a pdf (see §3.2). Make sure to justify all your answers.

1. **Experimental Design** (5 points)
   Describe how you set up your experiments to investigate what sets of features are useful for predicting the outcome of debates on **religious topics** vs. **debates on all other topics**.
2. **Performance**. (5 points)
   Show the accuracy for all models, including a simple majority baseline. Show all results in a single table.
3. **Analysis** (20 points)

[4] https://jsonlines.org/

(a) What two linguistic features did you incorporate in the *Ngram+Lex+Ling* model? State your hypothesis about which linguistic features you think would be useful (and why), and perform some data analysis to show that they would be useful to incorporate into your model. Give two examples where these features would help you identify the winner of a debate.

(b) What lexicon did you chose to incorporate and why? How did you define features using this lexicon? Perform some data analysis to show that the features you extract would be useful for predicting the winner. Give two examples where this feature would help you identify the winner of a debate.

(c) What user information do you think would be helpful in determining the winner of a debate and why? How will you incorporate this information as features into the model? Perform some data analysis to show that these features would be useful for predicting the winner of a debate. Give two examples where this feature would help you identify the winner of a debate.

(d) Did you find that different sets of features were useful for predicting the winner of a debate on religious topics vs. all other topics? Why do you think that's the case? What do your findings suggest about the factors that affect the outcome of a debate?

4. **Perpexity** (10 points).
You are given a training sequence of 100 animals that consists of 91 *badger*s and 1 each of 9 other animals (including a *snake*). You know that each non-*badger* occurs after ten *badger*s and the training sequence ends in a *badger*. Now, using add one smoothing, compute the bigram perplexity for each of the following test sequences where **B** denotes a *badger* and **S** denotes a *snake*. (Please ignore start-of-sequence and end-of-sequence characters).

   - **B B B B B S B B B B**
   - **B B S S B B S S B B**
   - **B B B B B B B B B B B B S**

5. **Smoothing** (10 points).
Consider the following corpus of text:

I went to the train station in San Francisco
I went to the bus station in San Francisco
I went to the metro station in San Francisco
Francisco went to San Francisco for gas

Suppose you build a bi-gram language model, with Katz backoff and absolute discounting with $d = 0.75$.

(a) What is the most likely completion for the sentence below from following set of words {station, Francisco, to}? Calculate the probability of the full sequence. Show your work.
I went to the gas _____

(b) Is what you get above consistent with what you would expect to be the most probable completion? If not, can you suggest a different smoothing method that could fix this issue?

3 Deliverables and Submission Instructions

Please read:

- **We WILL NOT accept code that does not run in Python 3.6.***, this includes broken code because of Python 2 print errors.
- **We WILL NOT accept hand-written work** for the written portion.
- File names (including for the zip file) must be **exactly** as specified for full credit

3.1 Programming (50 points)

Submit **one** zip file name **<YOUR-UNI>_hw1.zip** to CourseWorks.
This should have **exactly** the following files. Please DO NOT include the data files in the zip.
NOTE: your zip file should have only the following files:

- **`features.py`**: A file that takes a .json file containing the debates, and user.json file containing user information, and returns input features along with the labels. Indicate with comments the code for implementing each feature. If we cannot easily find the feature extraction, we will not grade it.
- **`hw1.py`**: A file that does the following

  - Takes 6 command line arguments: 1) the training file full path, 2) the testing file full path, 3) the full path to the user data, 4) the full path to the directory containing the lexica, 5) a model name (as a string), and 6) the full path to the output file. While working on this homework, you will pass the development file as the test file. NOTE: all this should be already done by the skeleton code provided and should not be modified.
  - Trains the model on the training set
  - Evaluates the model on the input testing file.
  - Writes the predictions to the output file.
    * Each line in the output file should contain the prediction of your trained model (i.e. Pro or Con) for each corresponding line in the test file. *NOTE: You may want to take into account the category of the debate in making your prediction.*

- **A README file that must include:**
  - Your name and email address.
  - Homework number.
  - Information on how to train and test your classifier.
  - A description of special features (or limitations) or your classifier.

Your code should be documented in a meaningful way and implemented efficiently. This can mean expressive function/variable names as well as commenting.

Your code should be runnable with the following command (as an example):

```
hw1.py --train <path_to_resources>/data/train.csv
    --test <path_to_resources>/data/dev.csv
    --model "Ngram+Lex"
    --lexicon_path <path_to_resources>/lexica/
```

where resources is the unzipped `resources.zip` file provided to you.

3.2 Written Answers (50 points)

You should submit the following on Gradescope:

- A **hw1-written.pdf** file containing your name, email address, the homework number, and your answers for the written portion.

4 Academic integrity

Copying or paraphrasing someone's work (code included), or permitting your own work to be copied or paraphrased, even if only in part, is not allowed, and will result in an automatic grade of 0 for the entire assignment or exam in which the copying or paraphrasing was done. Your grade should reflect your own work. If you believe you are going to have trouble completing an assignment, please talk to the instructor or TA in advance of the due date.

References

[1] Esin Durmus, and Claire Cardie. 2018. Exploring the Role of Prior Beliefs for Argument Persuasion. *Association for Computational Linguistics-2018*. `https://aclanthology.org/N18-1094/`

[2] Song Feng, Kang, J.S., Kuznetsova, p., and Choi, Y. 2013. Connotation Lexicon: A Dash of Sentiment Beneath the Surface Meaning. *Association for Computational Linguistics-2013*. `https://aclanthology.org/P13-1174/`

[3] Saif Mohammad. 2018. Obtaining Reliable Human Ratings of Valence, Arousal, and Dominance for 20,000 English Words. *Association for Computational Linguistics-2018*. `https://aclanthology.org/P18-1017/`