CT5103 Case Studies in Analytics: Assignment 2

Classifying Movie Reviews: Crowd-powered Machine Learning

Dr. Umair ul Hassan

Lecture: 08/02/2019, 09:00 GMT
Tutorial 1: 13/02/2019, 14:00 GMT
Tutorial 2: 20/02/2019, 14:00 GMT
Deadline: 21/02/2019, 23:59 GMT
Room: IT102
Questions to: niki.pavlopoulou@insight-centre.org

Assignment Goal & Dataset:

The assignment requires you to build models for classifying the polarity (i.e. positive or negative sentiment) of sentences in movie reviews. The assignment is based on movie reviews collected from the Rotten Tomatoes website[1] and polarity labels provided by crowd workers in Amazon Mechanical Turk. The dataset provided for this assignment contains three CSV files, as described below:

- The **gold-standard dataset** in file "gold.csv" which is based on the polarity of the reviews (i.e. positive=fresh and negative=rotten) in the website. The file contains 5000 rows (one per sentence) and 1202 columns. The first column is identifier for a sentence and last column is the polarity class (positive/negative). The remaining 1200 columns are features extracted from each sentence using Latent Semantic Analysis (LSA).

<id, TOPIC0,…,TOPIC1199,class>

- The **test dataset** is in file "test.csv" which is similar to the **gold-standard dataset** but contains 5428 rows (sentences different from the ones in gold-standard dataset) and 1202 columns.
- The **crowdsourced dataset** is in file "mturk.csv" which contains the polarity labels for sentences as given by crowd workers. The file contains 27,746 rows and 1203 columns. The extra column (i.e. annotator) identifies the crowd worker. The remaining columns are similar to **gold-standard** and **test datasets**.

< id, annotator, TOPIC0,…,TOPIC1199,class>

You are required to write code building three models for classifying the polarity of test dataset. First model will be based on the **gold-standard dataset** and the other two models are based on **crowdsourced dataset**. You will measure and compare the accuracy of all models using the **test dataset**. The overall goal is to understand the use of crowdsourcing for collecting labels that approximate the true polarity of sentences.

[1] https://www.rottentomatoes.com/

Assignment description:

The assignment must be completed in **Python 3 or Java** and following tasks are to be completed for this assignment. Code readability and proper comments would have marks.

Task	Description	Submission File
Part 1 - model training using gold-standard data		
1	Write a script to extract a random sample of 1000 rows from gold-standard dataset and save it to a file (i.e. gold_sample.csv)	**gold_sample.py** **gold_sample.csv**
2	Write a script to build a classification model using Decision Trees that take as input the file produced in task 1 (i.e. gold_sample.csv). The script should train a model, test its performance using test dataset (i.e. test.csv) and save the resulting model F-score, accuracy and prediction probabilities in a text file (i.e. train_gold.txt).	**train_gold.py**
3	Run the script from task 2 on your gold_sample.csv and submit the resulting file.	**train_gold.txt**
Part 2 - model training using crowdsourced data with majority voting		
4	Extract rows from crowdsourced dataset that correspond to the sentence ids of step 1 and save it to file (i.e. mturk_sample.csv)	**mturk_sample.py** **mturk_sample.csv**
5	Write a script to build a classification model using Decision Trees that take as input the file produced in task 4 (i.e. mturk_sample.csv). Aggregate the labels provided by workers using **majority vote** that assigns a point mass to the label with the highest consensus among a set of judgments. The script should aggregate labels, train a model, test its performance using test dataset (i.e. test.csv) and save the resulting model F-score, accuracy and prediction probabilities in a text file (i.e. train_mv.txt).	**train_mv.py**
6	Run the script from task 5 on your mturk_sample.csv and submit the resulting file.	**train_mv.txt**
Part 3 - model training using crowdsourced data with David & Skene method		
7	Write a script to build a classification model using Decision Trees that take as input the file produced in task 4 (i.e. mturk_sample.csv). Aggregate the labels provided by workers using **David & Skene** method that uses maximum likelihood estimation approach. The script should aggregate labels, train a model, test its performance using test dataset (i.e. test.csv) and save the resulting model F-score, accuracy and prediction probabilities in a text file (i.e. train_ds.txt).	**train_ds.py**
8	Run the script from task 7 on your mturk_sample.csv and submit the resulting file.	**train_ds.txt**
Part 4 - data description and results comparison		
9	Describe the characteristics of two samples extracted in task 1 and task 4 including total number of labels, distribution of labels, average number of workers per sentence, etc. Use graphs or tables to show the characteristics apart from the description. Explain your random sampling method and your reasons behind that. Explain how you handled ties if any.	**report.pdf**
10	Describe the 3 models and the methods used in step 3, 6, and 8 and compare their performance. Explain which model should behave the best. Discuss reasons for differences in performance. The discussion	

	should be based on the samples created in task 1 and task 4, as well as the methods used. The samples should be different from other groups. Use graphs or tables and confusion matrices to show the differences in performance.	

Marking Scheme:

- Code, data, and models: 50%
- Summary Report: 50%
- Groups of 2 students are the same as in Assignment1.
- Marking will be at group level provided that the contribution for both students is equal.
- Code, output files, and reports should be submitted for each group.

Submission Instructions:

- Please put your code, output files and report into a single .zip archive with name "Student1Name_Student2Name_CaseStudyAssignment2.zip" and submit via Blackboard
    - The folder structure of submission should be
        - Assingment2
            - report.pdf
            - scripts/
                - gold_sample.py
                - mturk_sample.py
                - train_gold.py
                - train_mv.py
                - train_ds.py
            - data/
                - gold_sample.csv
                - mturk_sample.csv
            - results/
                - train_gold.txt
                - train_mv.txt
                - train_ds.txt
- Include all source code files required to compile and run the code. Use relative paths for reading dataset files (.csv file) from the data directory. You do not have to include the initial three CSV files in the data directory in your submission, but the code should run as if they were there. In short, your submission should run from one directory (i.e. scripts folder) without the need to resolve dependencies and file paths for datasets. Incorrect file paths and code with errors will lead to mark deductions.
- You may use existing libraries for the Decision Trees, but the David & Skene method should be implemented from scratch.
- Use comments to explain your source code. Insufficient comments can lead to mark deductions.
- Add author names to source code files and report for identification. Files with missing identification can lead to mark deductions.

- The report should not consist of extracts or comments of the code, but it should only emphasise on the code results.
- Please note that all submissions (both code and report) will be checked for plagiarism.