Text Clustering/Classification Plugin for Web Server

Team: Abot

Aaron Botelho (botelho3@illinois.edu)

Importance

With the growing popularity of websites serving user generated content, discovery of interesting content among the millions of hours of video or millions of pages of written posts and reviews becomes difficult. The growth of forums like Reddit, design websites like Pinterest, interest based websites like Cooking.com, or reviews on any of the major retailer websites like Amazon.com have been explosive. Many such large sites rely on their large community of users to categorize their content. E.g Reddit with subreddits, Amazon with product categories, Cooking.com by cuisine, ingredients or author.

Smaller websites without the engineering resources to build such categorization systems are at a disadvantage. Without a large community of users to manually organize or classify contents into subcategories for users to navigate, or to tag/like content allowing classification models or user-similarity based recommender systems to be built. Operators of niche websites continue to lag behind their larger peers in offering these advanced features. Without such features users of smaller sites may become frustrated with the inconvenience and inability to find content of interest. A vicious cycle.

To remedy this issue of lack of investment relative to their larger peers. Smaller developers and webmasters often rely on open source tools, e.g Apache Lucene or ElasticSearch to provide search functionality to their website. In my experience smaller websites I use often have search capability thanks to Lucene or ElasticSearch, but are often lacking when it comes to content classification.

Task Summary

I plan to develop a content discovery and/or classification plugin for a common webserver to help remedy this functionality gap. As a proof of concept I'll implement either content discovery/clustering or classification functionality for generic user generated documents e.g. forum threads, reviews, or recipes. Automated content discovery may be done using PLSA or clustering via a simple mixture model. Classification functionality may be implemented with a Naive Bayes classifier or Logistic Regression.

Tools & Languages:

Language: Python

Tools:

- Webserver - Flask (Python)
- Numpy/Scipy
- May use other numerics packages for e.g classification

Datasets:

- Dataset of user generated content (Kaggle, Scraped etc.)
  - Amazon reviews
  - Film/Game reviews
  - Interest forums posts
  - Recipes/Food blogs.
  - E.g. https://www.kaggle.com/shuyangli94/food-com-recipes-and-user-interactions/version/2?select=RAW_recipes.csv

Approach

The first step is to find an example dataset on which to run the proposed clustering or classification algorithm. For classification there is the additional challenge of finding a dataset with tags or category labels included so that manual classification is not necessary. The dataset may also need to be cleaned (null removal, drop unnecessary columns) and split into a test, train set.

Once the dataset for the demo has been selected, work can begin on coding the algorithm needed to perform the classification or clustering of the dataset. For classification we propose Logistic Regression or a Naive Bayes Classifier. For clustering PLSA or a Simple Mixture Model. For either task the algorithms implemented will be the same as those covered in the course. The algorithm will be tested on the dataset until we are satisfied with the accuracy.

With the final accurate model developed the code needed to clean the dataset, read the dataset, initialize model parameters, iteratively or analytically solve the model, and define the result format will be formalized into a Flask plugin. The plugin is expected to consist of a few Python source files, some config files, an example dataset, and documentation.

Lastly in order to be able to demonstrate the results of our plugin a simple web interface will be developed to filter the dataset based on the clustering or classification results. This will likely be text based and will focus on returning the most important results if the dataset is large. If the first 3 steps take longer than expected we may forgo the web-interface in favor of a command-line python interface.

Workload

Task	Hours	Hours Cumulative
Collection + Cleaning of Dataset	6	6
Design of Classification or Clustering Algorithm	10	16
Coding of Plugin/Framework Implementing Algorithm in Flask	10	26
Designing basic web-interface to Demo results on Example dataset.	6	32
Documentation	2	34