Wilfrid Laurier University
Department of Physics and Computer Science

CP423A — Text Retrieval & Search Engine — Winter 2023

3^{rd} Assignment

Overview: This assignment includes four questions to cover a range of machine learning topics from the textbook.

What to submit? For the assignment, each question explains its specific deliverables. Also, you need to submit a **report.pdf** file that includes group members information, details of each group member contribution, and also your thorough explanation for each of questions. Finally, you are required to upload the **A3.zip** file into MyLearningspace drop box by the end of **Monday 3rd April**. Only one of group members should submit group's deliverable. Therefore, arrange it internally. Please avoid redundant submissions for a group.

Question-1(50 points). Sentiment classification is a technique used in text mining and machine learning (ML) to analyze and understand the emotional tone of a piece of text. This technique involves training a ML model using a dataset of labeled text, where each text is associated with a positive, negative, or neutral sentiment. The model then uses this training to classify the sentiment of new text inputs. Sentiment classification can be applied in various fields, including social media monitoring, product review analysis, and customer service feedback analysis. The aim of sentiment classification is to provide insights into the feelings and opinions of people towards a particular topic or product, and to help businesses make informed decisions based on this information. In this question, please download the "Sentiment Labelled Sentences Data Set" from the UCI repository. The corpus includes three files collected from IMDB, Yelp and Amazon users' reviews. Each file includes several records, and each includes a text and a number which indicates the text sentiment's label. Any real-world ML pipeline includes two fundamental procedures, training, and inference.

Therefore, this question includes the following steps:

STEP-1:

Write a `training_sentiment.py` that is callable from command prompt like so:

```
python training_sentiment.py [options]
```

a) `[Options]` are as below:

1. `--imdb`: it means IMDB is one of the datasets that is used for training ML models
2. `--amazon`: it means Amazon is one of the datasets that is used for training ML models
3. `--yelp`: it means Yelp is one of the datasets that is used for training ML models
4. `--naive`: it means naïve bayes is the classifier to train
5. `--knn [k]`: it means KNN using k-nearest neighbor is the classifier to train. The k is an integer positive number entered by user.
6. `--svm`: it means SVM is the classifier to train
7. `-decisiontree`: it means decision tree is the classifier to train

The script should load dataset(s) (option 1-3), tokenize and remove stopwords using NLTK library, train the selected classifier (option 4-7), report (print) performance of classification in terms of Accuracy, Recall, Precision and F-Measure, Plot confusion matrix and finally save the model. For training the classifier, you can use NLTK or scikit-learn libraries. For evaluating the performance of model, you need to apply Cross validation technique.

STEP-2:

Write a `predict_sentiment.py` that is callable from command prompt like so:

```
python predict_sentiment.py “text to classify”
```

The script should tokenize and remove the stopwords from the inputted text using NLTK library similar to step-1, load the saved classifier from the previous step, predict and print the label for the text.

Question-2(50 points). Clustering is a powerful technique in machine learning that can be used to group similar items based on their features. In the case of textual content, clustering can be applied to group together documents that share similar topics, themes, or styles. This technique is particularly useful when dealing with large volumes of unstructured text data, such as news articles or social media posts. One example of applying clustering on textual content is the 20-newsgroup dataset. This dataset contains thousands of newsgroup posts on various topics, such as politics, sports, and technology. To apply clustering on this dataset, we first need to preprocess the text by removing stop words, punctuation, and other irrelevant information. We can then represent each document as a vector of features, such as word frequency or term frequency-inverse document frequency (TF-IDF). Next, we can apply a clustering algorithm, such as k-means or hierarchical clustering, to group together similar documents. The number of clusters can be determined using techniques such as elbow method or silhouette analysis. Once the clustering is done, we can then analyze the resulting clusters to identify common topics or themes. For example, we might find that the documents in one cluster are all related to politics, while the documents in another cluster are related to sports. This information can be used to improve search engines, recommend related articles, or identify trending topics in social media.

Write cluster_news.py that is callable from command prompt like so:

```
python cluster_news.py [options]
```

a) `[Options]` are as below:

1. `--ncluster [n1,n2,n3,…]`: This parameter tells your script about the number of cluster(s). If we have more than one number of clusters, it means you should repeat your clustering for each number. Default value is 20.
2. `--kmeans`: it means you need to use KMeans clustering.
3. `--whc`: It means you need to use Ward Hierarchical Clustering clustering.
4. `--ac`: It means you need to use agglomerative clustering clustering.
5. `--dbscan`: It means you need to use DBSCAN clustering.

The script loads and preprocess dataset as explained above, cluster items and print performance. For printing your performance, you should report Adjusted Mutual Information, Adjusted Random Score, and Completeness score. Obviously, if you need a ground truth label, you can assume that all the records of a news group has same labels. For instance, you can set ground truth label of items in "sci.space" as 0, "rec.autos" as 1 and so on. Finally, you should save your clustering model.

If you have more than one cluster number, you should repeat clustering and evaluation and for each cluster number report your performance.

*** for clustering, only use main textual content of each file and ignore date, sender, path, organization and etc.