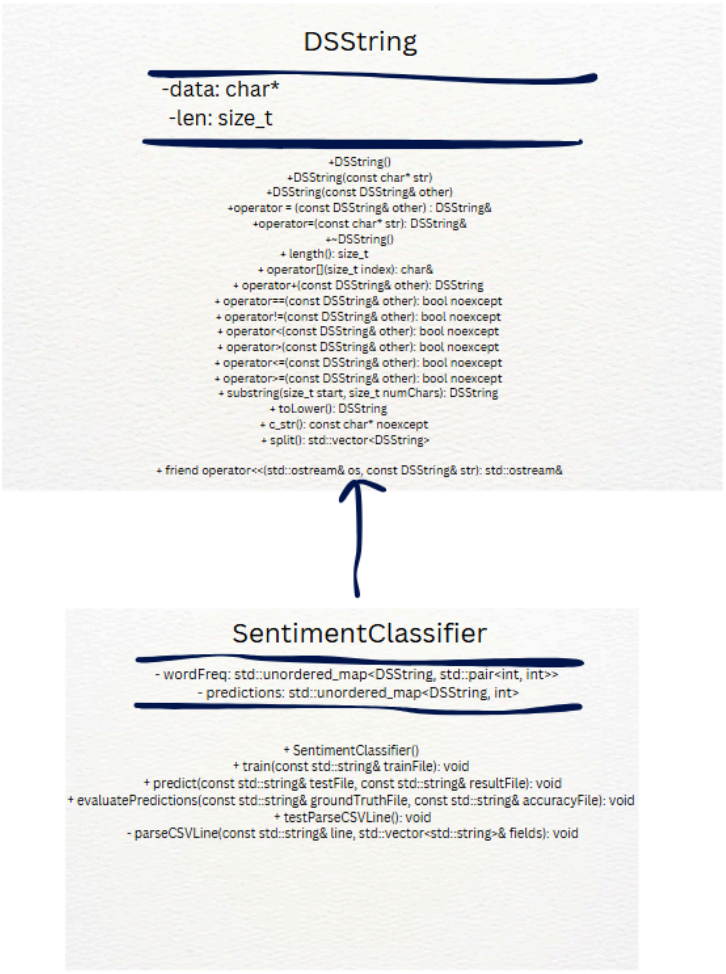
Name: Nimai Keshu
Collaborators: N/A
Date: 11/4/24
CS5393-002-1247

Note: I used **./sentiment data/train_dataset_20k.csv data/test_dataset_10k.csv data/test_dataset_sentiment_10k.csv output** to run the program
Sentiment Classifier - Project 3
Github: https://github.com/nkeshu/project-3/tree/main
UML Diagram:

UML Diagram



Outline of the code:
This project creates a basic sentiment analysis classifier for given tweets. The main goal is to predict if a tweet expresses a positive or a negative sentiment based on the words. It uses training data to build a model of word frequencies that are associated with either positive or

negative sentiments. Then, it uses that model to predict sentiments for new tweets in a test data set. It will evaluate the accuracy at the end. For the DSString class, I used several different functions to create my own string class without using std::string. In the SentimentClassifier class, it contains the most important functions of the analysis process which includes training, predicting, and evaluating. It had one private method that parsed the CSV line into individual fields. In the main function, I used functions to train/make predictions / evaluate the results. There are a lot of functions in this project, the most important functions of the sentimentclassifier class include the train function which has a parameter of the trainfile. It reads the training data file and then processes each tweet in wordfreq. The predict function takes in the testfile /result file as parameters, and then it will predict the sentiment and write the predictions to the result file. The evaluatepredictions function takes in the groundtruthfile and the accruacyfile files and it will calculate accuracy + put down misclassified tweets in the accuracy file.

1. How do you train the model and how do you classify a new tweet? Give a short description of the main steps.

To train the model you need to parse the test data first by opening the test file and skipping the header line. Then you process every line and then extract the id and the text. After that, you convert to lowercase and then tokenize the tweet. Then you compute the sentiment score, predict the sentiment, and output predictions to the results file.

2. How long did your code take for training and what is the time complexity of your training implementation (Big-Oh notation)? Remember that training includes reading the tweets, breaking it into words, counting, ... Explain why you get this complexity (e.g., what does N stand for and how do your data structures/algorithms affect the complexity).

The time complexity of training is O(N*L). N is the number of tweets in the training dataset and L is the average number of words per tweet. Reading the tweets is O(N), it processes the tweets

one by one. Parsing and tokenizing the tweets is O(L) per tweet where L is the number of words in the tweet since it parses the lines to extract the tweet, converts the tweet to lowercase, and then splits it into tokens. And then updating word frequencies is O(1) per word, all together its O(NXL).

3. How long did your code take for classification and what is the time complexity of your classification implementation (Big-Oh notation)? Explain why.

The time complexity for the classification is O(M(*L') where M is number of tweets and L is the average number of tweets per tweet. Reading tweets is O(M). Parsing / tokenizing is O(L') per tweet. All together its O(MxL').

4.What accuracy did your algorithm achieve on the provides training and test data?

My accuracy was a 73.4%

5.What were the changes that you made that improved the accuracy the most?

Correcting the parsing function of the file really helped me out and also implementing a way to exclude the header.

6.How do you know that you use proper memory management? I.e., how do you know that you do not have a memory leak?

All the allocated memory in the DSString class was deallocated preventing memory leaks.

7.What was the most challenging part of the assignment?

The predict function was pretty tricky for me. After using different ways to approach it I finally got it after some help with GPT.

References

ChatGPT