Madison Jordan
Luke Duggan
Naoki Atkins

Author Classification using KNN

README.md

479-Project-2

Project of choice in data science

Group members:

Naoki Atkins naokishami@csu.fullerton.edu

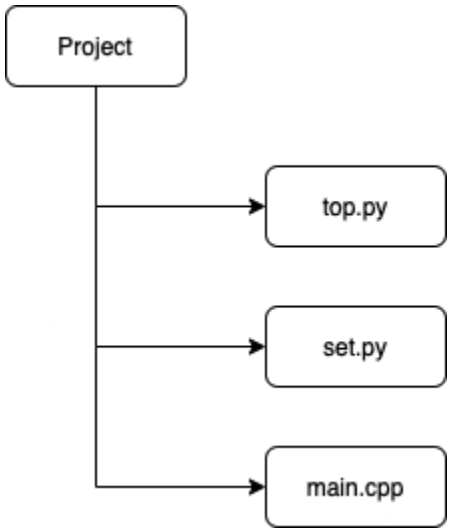
Luke Duggan luke.duggan@csu.fullerton.edu

Madison Jordan madisonjordan@csu.fullerton.edu

Summary

In this project, we created a program that will classify an unknown book to an author given a dataset that contains already known books with labels to their authors.

1. In the preprocessing stage, we cleaned “stopwords” out of the book txt files. These “stopwords” are words like ‘a’, ‘the’, ‘their’, ‘what’, etc.
2. After ridding the books of these “stopwords”, we found the top 100 most used words from each book with their frequency count using ***top.py***.
3. Then we ran these frequency counts into ***set.py*** to find a union of all these words and combined it into a single table.
4. Thereafter, we ran ***main.cpp*** to calculate the distance between a query book and each known book and finally classifying this query book using the majority of the k nearest neighbors.



Pseudocode

Text Preprocessing

1. Create a bag of words from each known text
    - Remove most common “stopwords” eg) “the”
2. Find the frequencies of the unique words in each text
    - find top 100 in each, if necessary. to reduce dataset size
3. Create a matrix formed from the union of the words in each text
    - Frequencies filled as 0 for words that do not appear in that particular book

KNN

4. Find the cosine similarity distance from the unknown book to all known books
    - Parallelize the calculation of each of the word frequencies in the book
5. Find the “k” closest books (neighbors)
6. Get the labels of the k nearest neighbors
7. Find the label that represents the greatest majority of the labels in the k nearest neighbors. This is the predicted label of the unknown book.

How to Run

Text Preprocessing

run each time a new dataset needs to be generated (if a new book is added).

- input folder (book text files): `\books`
- output folder (dataset files): `\data`

First run top.py to find the "bag of words" from each book. After that, you'll need to run set.py so that each "bag of words" can be combined into a single table.

- `python3 top.py`
- `python3 set.py`

Compiling

```
mpic++ -fopenmp main.cpp
```

Run Program

```
mpirun -n 4 ./a.out
```

note: n processes should be the number of books in the set. (used in cosine calculation before finding KNN). k is a separate value from n. k is defined within the code at this time (k=3)

Code Screenshots

No Majority

```
win_ubun@Scheherazade:~/projects/hpc/projects/branch/main$ mpirun -n 4 ./a.out
greatgatsby cos similarity to querybook: 0.191499
mobydick cos similarity to querybook: 0.279757
romeoandjuliet cos similarity to querybook: 0.378174

Showing only the nearest k = 3 nearest neighbors...

nearest neighbor 1
        | Rank 0 = 0.378174
        | Book (romeoandjuliet) has label 3

nearest neighbor 2
        | Rank 2 = 0.279757
        | Book (mobydick) has label 2

nearest neighbor 3
        | Rank 1 = 0.191499
        | Book (greatgatsby) has label 1

RESULT: using k = 3,
        Predicted label class is 1 -- if no majority, picked one
```

Majority

```
win_ubun@Scheherazade:~/projects/hpc/projects/branch/main$ mpirun -n 4 ./a.out
mobydick cos similarity to querybook: 0.279757
greatgatsby cos similarity to querybook: 0.191499
romeoandjuliet cos similarity to querybook: 0.378174

Showing only the nearest k = 3 nearest neighbors...

nearest neighbor 1
        | Rank 0 = 0.378174
        | Book (romeoandjuliet) has label 3

nearest neighbor 2
        | Rank 2 = 0.279757
        | Book (mobydick) has label 3

nearest neighbor 3
        | Rank 1 = 0.191499
        | Book (greatgatsby) has label 1

RESULT: using k = 3,
        Predicted label class is 3 -- if no majority, picked one
```

Note: Changed label of MobyDick from “Melville” to “Shakespeare” to test majority function