Text Analysis in Classic Literature

Seth Doubek and Kimball Johnston

University of Utah

Acknowledgements: Professor Qingyao Ai and the CS5140/DS6140 Teaching Staff

ABSTRACT

For our final project, we explored text analysis by analyzing works of classic literature to identify clusters among them.

Specifically, we studied literary works whose text is freely available online, such as the Bible, the works of Charles Dickens, and *War and Peace.* Once we had assembled a collection of these texts drawn from a variety of authors and periods, we created a vector representation for each book using k-grams. We then combined these vectors as rows in a data matrix. This data had a very large number of columns, so we performed principal component analysis (PCA) on our data set to lower its dimensionality to a more reasonable level.

After reducing the data set's dimensionality, our final step was to identify clusters in the data. We used a variety of different clustering methods, such as Lloyd's algorithm and mixture of Gaussians, and then compared their results.

Data Collection

We collected our data from Project Guttenberg, which is an archive of freely-available, online books[1]. Thus, our dataset was limited to books whose copyrights had expired in the U.S. Our dataset consisted of 69 text files, which, for brevity we will not list here. See the "Results" section for the full list of books we used.

Text Vectorization

We vectorized our data with two techniques: word k-grams, as defined in the class's textbook [2], and the TF-IDF algorithm [4]. For k-grams, we randomly selected n k-grams (where n and k are parameters) from each book. We combined these n k-grams from each book into a single library of k-grams. Each book was represented by a vector whose entries corresponded to specific k-grams in our k-gram library. Each entry was 1 if that text file contained that k-gram and was 0 otherwise.

Dimensionality Reduction

We reduced the dimensionality of the vectors representing our text files to d (where d is a parameter) using PCA [2]. We did this by centering the data, running SVD on the centered data, and then finding the coordinates of the data's projection onto the d-dimensional basis formed by the top d right singular vectors.

Clustering

We identified clusters in our dimension-reduced data using three techniques: Lloyd's algorithm, the expectation maximization algorithm for mixture of Gaussians [2], and hierarchical agglomerative clustering (HAC). We decided to try out mixture of Gaussians because the data's nature lent itself to soft clustering (it seems likely that regardless of how you defined clusters, some books would fall into multiple clusters).

For each clustering technique, we tried out multiple parameter combinations, including different numbers of centers, different initialization techniques (Gonzalez's algorithm vs. k-means++), different levels of dimensionality reduction, and different text-vectorization levels (2-grams vs. 3-grams).

Evaluation

We evaluated our results based on two key factors: whether books with the same author were placed into the same cluster, and whether books in a cluster had similar genres.

Results

Lloyd's algorithm:

Based on our above evaluation criteria, the best results we found with Lloyd's algorithm occurred when we vectorized the texts using 2-grams, we reduced the data to 5 dimensions, we used 7 centers, and we initialized the clusters with k-means++. These clusters are reported at the top of the other half of the page.

We judged this set of clusters to be best because the clusters fell into good genre groups (with the exceptions of *Gatsby* in the children's books and the Odyssey in the Romantic books). Furthermore, almost all books by the same author fell into the same cluster. The two exceptions to this were *A Christmas Carol* (which is much shorter and different than Dickens's other entries on the list) and the works of Alexandre Dumas. Using Lloyd's algorithm, Dumas's works were often placed in different clusters, regardless of the parameters we used. This could, of course, be due to Dumas's works being translations. Upon further research, however, we discovered that Dumas was known for working with a variety of anonymous collaborators to write his books [3], so the fact that our algorithm separated these books is not necessarily a bad result.

0: Poems and Short Novels
Poems by Emily Dickenson
Poirot Investigates
The Scarlet Pimpernel
Aesop's Fables
A Christmas Carol
Heart of Darkness
Poems by Robert Frost
The Strange Case of Dr. Jekyll and Mr. Hyde
Treasure Island
The Jungle Book
The Mark of Zorro
The Rime of the Ancient Mariner
The Merry Adventures of Robin Hood

5: Epics
Moby Dick
Ulysses
War and Peace
Don Quixote
The Travels of Marco Polo Vol. 1
Les Miserables
The Bible
The Count of Monte Cristo
Anna Karenina

1: Romantic Literature and Literary Realism
The Adventures of Sherlock Holmes
Great Expectations
A Tale of Two Cities
The Three Musketeers
Crime and Punishment
The Importance of Being Earnest
Uncle Tom's Cabin
Dracula
Wuthering Heights
Jane Eyre
The Odyssey

4: Ancient Poetry
Canterbury Tales
1001 Arabian Nights Vol. 1
The Divine Comedy
Paradise Lost
The Iliad
Don Juan
The Koran
Beowulf
Macbeth
Hamlet
Romeo and Juliet

6: Jane Austen
Sense and Sensibility
Emma
Pride and Prejudice

2: Adventure and Romance
Around the World in 80 Days
The Man in the Iron Mask
A Journey to the Center of the Earth
Gulliver's Travels
Ivanhoe
The Scarlet Letter
Frankenstein
The Last of the Mohicans
Plutarch's Lives Vol. 1
Poems by Edgar Allen Poe
20000 Leagues under the Sea

3: Children's and Pulp
Peter Pan
The Great Gatsby
The Wonderful Wizard of Oz
Tarzan of the Apes
King Solomon's Mines
The Invisible Man
Grimms' Fairy Tales
The Adventures of Tom Sawyer
Adventures of Huckleberry Finn
Alice in Wonderland
The Time Machine

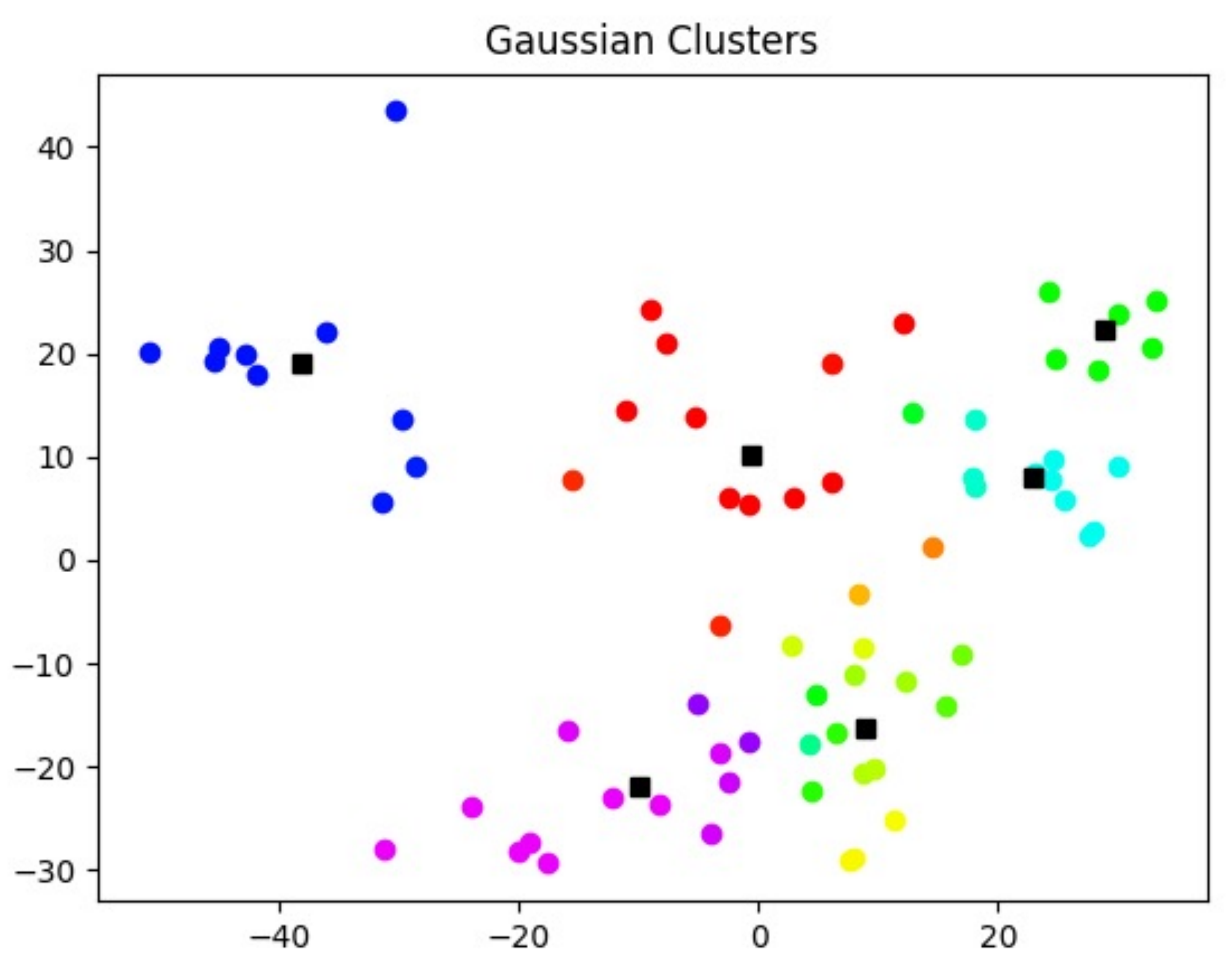

Figure 1: Mixture of Gaussians clustering on 2D data

Mixture of Gaussians:

Using mixture of Gaussians clustering, we found similar results to Lloyd's algorithm. A notable insight that mixture of Gaussians yielded was that *A Tale of Two Cities* and *The Three Musketeers* were often grouped both in clusters with 19th century high-brow literature like *Crime and Punishment* and in clusters with more pulpy works like *Around the World in 80 Days* and H.G. Wells.

HAC:

We experimented with using single linkage, complete linkage, and mean linkage for HAC. The resulting clusters were very similar to what was found using Lloyd's algorithm, with a few additional insights.

Firstly, HAC very often grouped Shakespeare's works together into their own category, similar to what happened with Jane Austen in our above clusters for Lloyd's algorithm. Secondly, unlike our final results for Lloyd's algorithm, HAC grouped the Bible together with the Koran, which we personally found interesting. Finally, we found that using single or complete linkage would often result in clusters that only had one or two books. *The Merry Adventures of Robinhood* and Emily Dickenson's poetry especially were often in these isolated clusters.

TF-IDF:

As mentioned in the text vectorization section, we experimented with using TF-IDF (Term Frequency—Inverse Document Frequency) for text vectorization. This technique is often used in projects like this one. However, based on our results, it performed worse than using k-grams, so our analyses above used k-grams instead of TF-IDF.

References

[1] *About Project Gutenberg*. Project Gutenberg. https://www.gutenberg.org/about/
[2] Phillips, J. M. (2019). *Mathematical Foundations for Data Analysis* (v0.6 ed.)
[3] Editors of Encyclopedia Britannica. *Alexandre Dumas, père*.
Encyclopedia Britannica. https://www.britannica.com/biography/Alexandre-Dumas-pere
[4] Anirudha Simha. (2021, October 6). *Understanding TF-IDF for Machine Learning.* Capitol One. https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/