Machine Learning Engineer Nanodegree

Capstone Proposal

Date: June 17, 2018

Proposer: Emily Behlmann

Subject of project: Book recommendation engine using unsupervised learning on book rating data

Domain Background

My proposed capstone project is to create an application that will recommend books to an individual based on the person's taste and the book ratings of fellow readers. I'm an avid reader, and I'm always looking for my next favorite book. Like many readers, I have an account on Goodreads, a popular website for tracking your reading, reviewing books and connecting with fellow readers. The site, which says that it has 75 million members, provides book recommendations that it states are based on 20 billion data points (Chandler). However, I've never been very interested in the books Goodreads has recommended for me. This made me wonder if I could create a better recommendation engine.

The need to find books suited to one's taste is not society's most pressing problem. However, people's leisure time is limited, so it's worth thinking about how best to spend it. The average adult can read about 250 words per minute (De Leeuw). That means a typical novel of 90,000 words would take a typical person 360 minutes, or 6 hours, of continuous reading time — a significant investment of time for something one is not enjoying. Traditionally, we've relied on experts like booksellers to provide us with good book recommendations, but bookstores are becoming scarce. The Census Bureau's County Business Patterns survey found 6,448 bookstore establishments operating in the United States in 2016, down from 13,136 in 1992. Meanwhile, the world of published books grows each year, making it increasingly difficult for readers to find the books they will most enjoy among everything else.

Problem Statement

The goal of my project is to help readers find books to read next that are aligned with their tastes. Although readers' opinions about books are subjective, I will be able to measure how well readers like or dislike books based on the ratings they give them on a five-star scale. A good solution will be one that maximizes the rating score readers give to the books selected for them.

Datasets and Inputs

I intend to build my book recommendation engine based on the goodbooks-10k dataset, which was published under a Creative Commons license by Zygunt Zajac on the FastML website in 2017. The dataset includes about 6 million ratings of 10,000 books with the highest volume of ratings (ratings.csv). Each rating is associated with a user ID, making it possible to cluster individual readers based on their ratings. The dataset also includes metadata about each book, such as its title, author and average rating (books.csv).

Solution Statement

I plan to solve this problem by applying a clustering algorithm to the users who have rated books in the dataset. This will have the effect of creating clusters of readers who have similar taste in books. Then, when an individual reader has submitted ratings, I will be able to associate that user with the cluster of best fit and recommend books that the person hasn't read but that other members of the cluster rated highly.

Benchmark Model

The objective of the book recommendation engine is to recommend books a person hasn't read. However, evaluating the quality of these recommendations will be difficult without the opportunity to wait for readers to read the book and return with feedback. Therefore, I will need to take a different approach when comparing the quality of my recommendations to those of a benchmark model.

I plan to set aside a set of users as test data. I will apply my model and a benchmark model to each user in the test set. In the case of my model, I will cluster each user. Then, for books the user has already read, I will calculate the difference between the user's ratings and the cluster's average ratings. For the benchmark model, I will calculate the difference between the user's ratings and the dataset-wide average ratings. My expectation is that the user's ratings will more closely match the ratings of his or her cluster than those of the entire population of users.

Evaluation Metrics

I plan to do multiple evaluations over the course of the project — first to select the appropriate clustering model, then when choosing a k value (number of clusters) if my chosen model requires it, and finally when comparing the results of my model to the benchmark. I do not know ahead of time the "ground truth," or the number of reader clusters that exist in my dataset. Therefore, calculating the silhouette coefficient on various models and numbers of clusters will be an appropriate means of evaluation. The silhouette coefficient helps to illustrate how well points fit within their clusters based on two scores (Scikit Learn):

1. The mean distance between a sample and other points within the same cluster, and
2. The mean distance between a sample and all points in the next nearest cluster.

I will evaluate multiple clustering algorithms and multiple k values (where the model requires selecting a k value) and choose the one that results in the highest silhouette coefficient.

After I've determined my clusters, I plan to perform a final evaluation against a benchmark model as described in the section above.

Project design

I will begin my project by importing and assessing my dataset. To prepare my data, I will need to combine the ratings data and the book data into a single dataframe that has user IDs on one axis and books on the other, subtracting out a test set of users. Although my dataset includes only the 10,000 books with the most ratings, it will still be sparse, because there will be many books each user has not read. Therefore, I will assess whether I need to limit my analysis to the users who have completed the most ratings. If I wish to apply the k-means algorithm, I will need to cast my dataframe to a sparse csr matrix using the SciPi library, as was done with the MovieLens movie rating data in the Udacity mini-project "k-means clustering of movie ratings." Because of the high degree of dimensionality in my data (many users have rated many books), I will also consider applying principal component analysis. This analysis may identify composite features that represent phenomena within the data, such as a preference or dislike for a certain combination of genres.

Next, I will apply multiple clustering algorithms to the matrix and evaluate the silhouette coefficient for various permutations of each:

- k-means clustering with various k values.
- Gaussian mixture models with various k values.
- Hierarchical clustering methods such as single-link and Ward's method.

I will select the algorithm and parameters that maximize the silhouette score.

After selecting the best model and parameters, I will plot the clusters and make any observations I can about the results. I will then evaluate my results by clustering the users in my test set and determining how well-aligned their ratings are with their clusters' averages. Finally, I will develop a book recommendation function. The function's input will be a user's set of book ratings. The function will cluster the user, and its output will be a list of book recommendations based on the favorite books of the cluster that the user has not yet read.

Works cited

Chandler, O. *About Goodreads.* Retrieved from https://www.goodreads.com

De Leeuw, M., & De Leeuw, E. (1990). *Read better, read faster: A new approach to effecient reading*. London: Penguin.

Scikit Learn *Clustering performance evaluation: Silhouette Coefficient*. Retrieved from: Scikit-learn.org

U.S. Census Bureau (2018, April 19). *Geography Area Series: County Business Patterns 2016 Business Patterns.* Retrieved from https://factfinder.census.gov/

Zajac, Z. (2017, November 29). Goodbooks-10k: A new dataset for book recommendations. Retrieved from: http://fastml.com/goodbooks-10k