**Executive Summary Group 4**

Sabrina Alves, Ebrahim Moosa, Cody Gunter, Lyndah Mupfunya

The aim of our project is to create a machine learning recommendation engine that will allow people to get book recommendations based on a book that they enjoy. We are using the Goodreads Best Books Ever dataset from Kaggle to build our model off of. Our inspiration came by looking at the Anime Recommender that had been done by a previous bootcamp, as well as our own personal love for reading. When looking for data sources, we found two other book recommender systems that gave us ideas on how to tackle the problem. Our expectation from the model is that based on the recommendation model we use, the user will get 10 books that are similar to the input book.

Our original plan for the data was to use the Goodreads API to get live information, but as of December of 2020 the API has been closed. The dataset that we are using is the Goodreads Best Books Ever dataset from Kaggle. It consists of the first 10,000 books on the Goodreads best books ever list. The dataset in its base form has 12 columns and 10,000 rows of data. No cleaning was performed by the original poster of the dataset so duplicates and null values are still present in the dataset. The cleaning that took place on our end consisted of reducing the rows to just the information we needed to build the models. We reduced rows by removing duplicates, foreign languages, and null values. The removal of the ISBN column was done due to only 85% of the books having their number.

Two models were created using this cleaned data. The first model that was built was based on Natural Language Processing (NLP) to create a content based model. Using Rake (from the NLTK library), stopwords, punctuation, and white space were removed from the bookDesc and bookTitle column. All words were made lowercase and made a list for easier future processing. This did not remove all the symbols or numbers from the data so a separate function was created to remove the remaining characters. The data was then run through Bag of Words to tokenize the data and put into Bag of Word columns. Sentiment analysis was also run on the words using TextBlob to generate polarity and subjectivity scores. These scores were used on the KNN model that we will discuss later. The final step is to run a count vectorizer on the Bag of Word columns. A count vectorizer simply counts the frequency a word appears. TF-IDF was used to highlight less frequently used words such as ‘Hogwarts’. The TF-IDF scores produced will be a combined score. It will give higher scores to terms that occur frequently in a single/few document(s) than a term that frequently shows up across all documents. To compare the scores of each book, cosine similarity is used. Cosine similarity compares the score of two documents, so in our case the input book and the books that are closest to it in score. This is done by looking at the angle between each score in three dimensional space.

A KNN model was used for the second model. Unlike the NLP based model, KNN analyzes numeric data to create its recommendations. Book rating, number of rating, and review count were the top features for the model. Aspects of the NLP model were included in this model as well with inclusion of the polarity and subjectivity scores. One-hot encoding was performed

on the genre and author field to provide more data for the model and give a more curated recommendation. To give us an idea on how the model would recommend books, the data was plotted using TSNE to reduce the dimensionality of our data to just 2 features. This visual aid shows us that the books in the data set cannot be easily separated into clusters, apart from a few outliers. When the input book is entered, the KNN model looks for the 10 closest neighbors to the input point and those are the books it recommends.

Two Tableau dashboards were created to help visualize the dataset that we are using. Unlike the models, the dashboards provide more insight into the macro details of the dataset. Both dashboards filter their data using genre, either one or multiple can be selected at a time. The first dashboard presents the top 10 in the most reviewed books, the most reviewed authors, and the highest rated books. The second dashboard explores count of books by author, longest reads, and average page count per genre. Looking at these dashboards, we can find the most popular genres, if there are popular books within less popular genres, and which authors are the most successful.

We would like users to rely on our website for their book exploration needs. Whether they're exploring the most popular books by genre, want a book recommended based on content, or a book recommender based on popularity metrics, we hope we can build a one-stop shop for our users.