Online Book Retailer – Technical Write-up

Group 1: Derek K., Ian O., Patrick P., Treva W.

**INTRODUCTION**

An established book retailer, we are expanding into new markets by launching an online store. Our competitive advantage will be helping customers find the books they want and suggesting new books that may be of interest based on previous searches. Our books will be classified by reading level and inherent sub-categories. For this project, we are using the Project Gutenberg API as our library.

***getbooks()* FUNCTION**

Import data into the environment using our user defined *getbooks()* function. *getbooks()* takes a given numeric value (*x*) and outputs a dataframe with *gutenberg\_id*, *text*, *author*, *title*, and *subjects*.

After retrieving our initial dataset of x observations, the function filters out books with no text, non-english books, speeches, and congressional statements. Books without an assigned subject are removed and books that are very long or very short are moved. Books with UTF-8 encoding are filtered out so that text is appropriately formatted for readability model.

**DATA PREPARATION**

The hundreds of library subjects were manually reviewed and grouped to pair down the topic list to broader topics. For example, individual countries are grouped into “Geography” and multiple faiths are grouped into “Religion”.

**MODELING**

Within the *readability()* package, the Coleman Liau Index is used to calculate the average number of letters per one-hundred words and the average number of sentences per one-hundred words. The levels run from one to eighteen (second year graduate student).

Four models were built and tested for classifying subjects for the books using the *RTextTools* package. The models included SVM, Random Forest, Naïve Bayes, and Maximum Entropy. Other models were attempted, but were too computationally intensive to run. These included NNET, GLMNET, SLDA, BOOSTING, and BAGGING. After acquiring the books from Gutenberg and formatting them with the *getbooks()* function, the resulting data frame was put into the *create\_matrix()* function to create a document term matrix. The text was stemmed, punctuation removed, stop words removed, lower case, white space removed, and weighted with tf-idf weighting. The model results were not affected by stemming so the text was left stemmed. The matrix was then put into a container and split into a train and test set and the new condensed subject was used as the classifier in the *create\_container()* function. The three models were trained and then tested with the *classify\_model*() function on the test set.

A Naive Bayes Classification model was also ran using the e1071 package *naiveBayes()* function. The book subjects were coerced into factors for this model. The data was split into train and test sets. The train set contained 70% of the data, and the test contained the other 30%. A model was run on the train set, then predicted using the test set. Looking at the predictions, the model classified all the books into the same subject category (the category that was most represented in the data set).

The Naïve Bayes Classification model was not evaluated due to all the classifications being the same, when the books had varying subjects unrelated to the subject classified.

**EVALUATION**

Using the *create\_analytics()* function, it was determined that the Maximum Entropy was the best predictor of subjects. The F-Score, recall, and precision were all higher for the Maximum Entropy model than for the other models. The SVM model was the worst performing model.



**DEPLOYMENT**

These models will be applied to our inventory to recommend texts to enhance the user experience and increase add-on sales.