Samuel Nordale

Hafsa Khalid

Nicholas Paun

COMP 551 – Miniproject 2

**Abstract:**

The goal of the project was to develop a data model that could predict which subreddits a Reddit comment came from. This was after having learned on a dataset of Reddit comments and their associated subreddits. Our most interesting findings were x, y, z.

**Introduction:**

There were three core tasks assigned in the project. First, we were required to implement a Bernoulli Naïve Bayes model from scratch. Second, we were to run experiments on the performance of several classification models from Sci-Kit Learn. Lastly, we needed to create a model validation pipeline and report on the differences between the models.

The Reddit comment dataset that we received included 70,000 comments from 20 different subreddits, and the fields defined in the table were the comments’ text, subreddits, and ID number.

The purpose of implementing Bernoulli Naïve Bayes was to represent the documents as binary vectors, where a 1 indicated that a specific word occurred in the document, and a 0 indicated the absence there of. As a result, the class-likelihood of a particular document was modelled using a Bernoulli distribution, giving the name Bernoulli Naive Bayes, This is what what we implemented from scratch.

**Dataset and Setup:**

The biggest factor when processing the dataset was the fact that the comments did not conform to a common standard for language, format, or punctuation. Some comments had unique formatting for different types of information due to the standard of the specific subreddit. Some comments used links, emoticons, and emojis, which need to be accounted for.

**Proposed Approach:**

We implemented our own Multinomial Naïve Bayes model, and to compare our results with different classification models from Sci-Kit Learn, we chose: Logistic Regression, Multinomial Naïve Bayes, and Random Forest. Our feature design focused on creating arrays of keywords for subreddits. Then, by comparing each comment to the keywords, we could signal how likely it was that an individual comment was from a specific subreddit. Other features that we implemented were comment length, word count, and average sentence length. We hoped that this would help the model identify subreddits where users engaged in more long-form discourse.

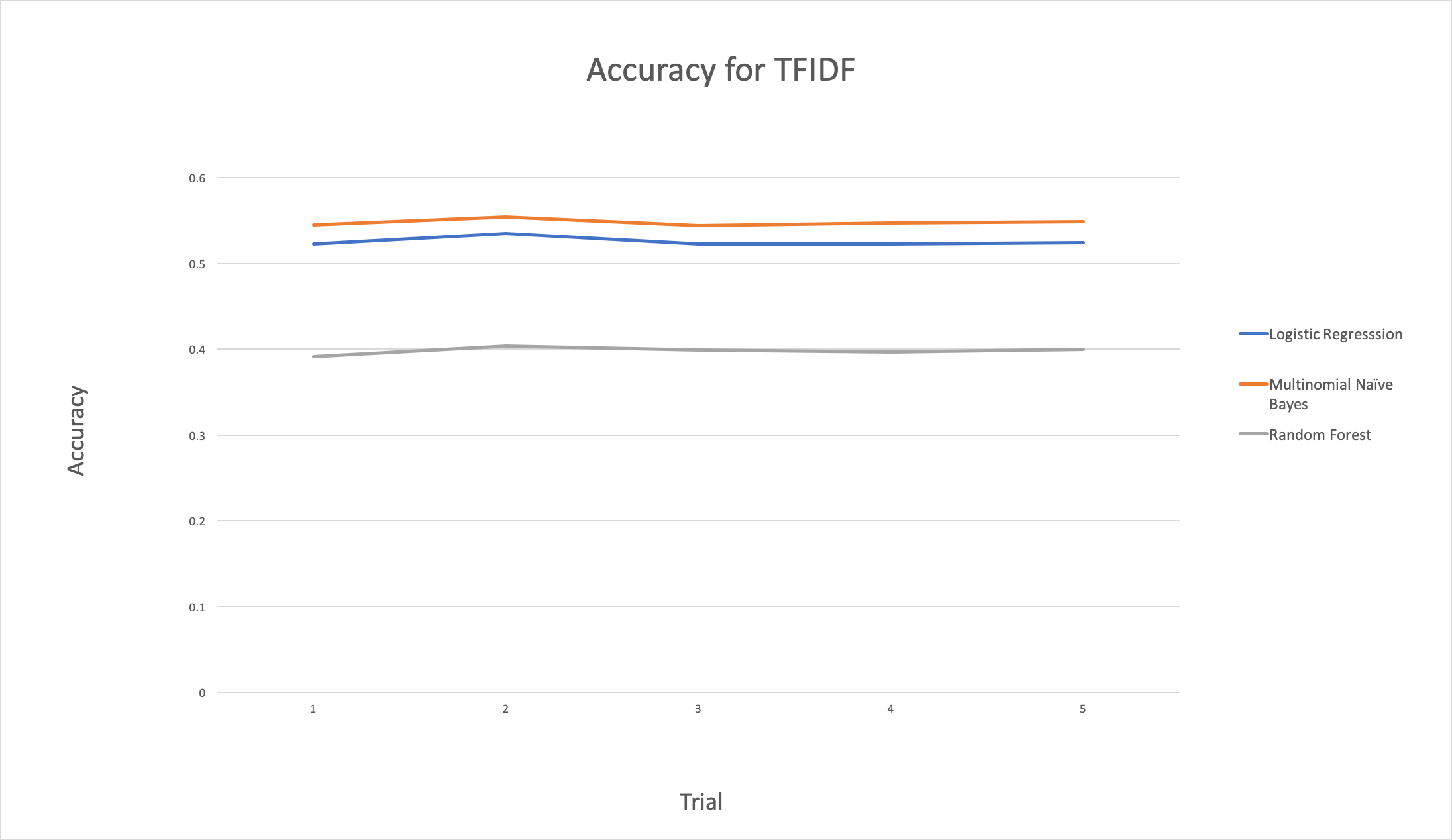
For our Multinomial Naïve Bayes experiment, we used a unigram model with English stop words and no additional constructed features were chosen, as prior results obtained using Sci-Kit Learn suggested that more complex features did not greatly improve performance. To handle out-of-vocabulary items, Laplace smoothing was performed on the training data. In addition, to avoid the risk of exteremely small probability values leading to error, calculations were performed using log odds.

**Results:**

As our baseline, we calculated the accuracies for Sci-Kit Learn’s TF-IDF and Count Vectorizers. Using just the feature matrix given by the vectorizers, we calculated these accuracies:

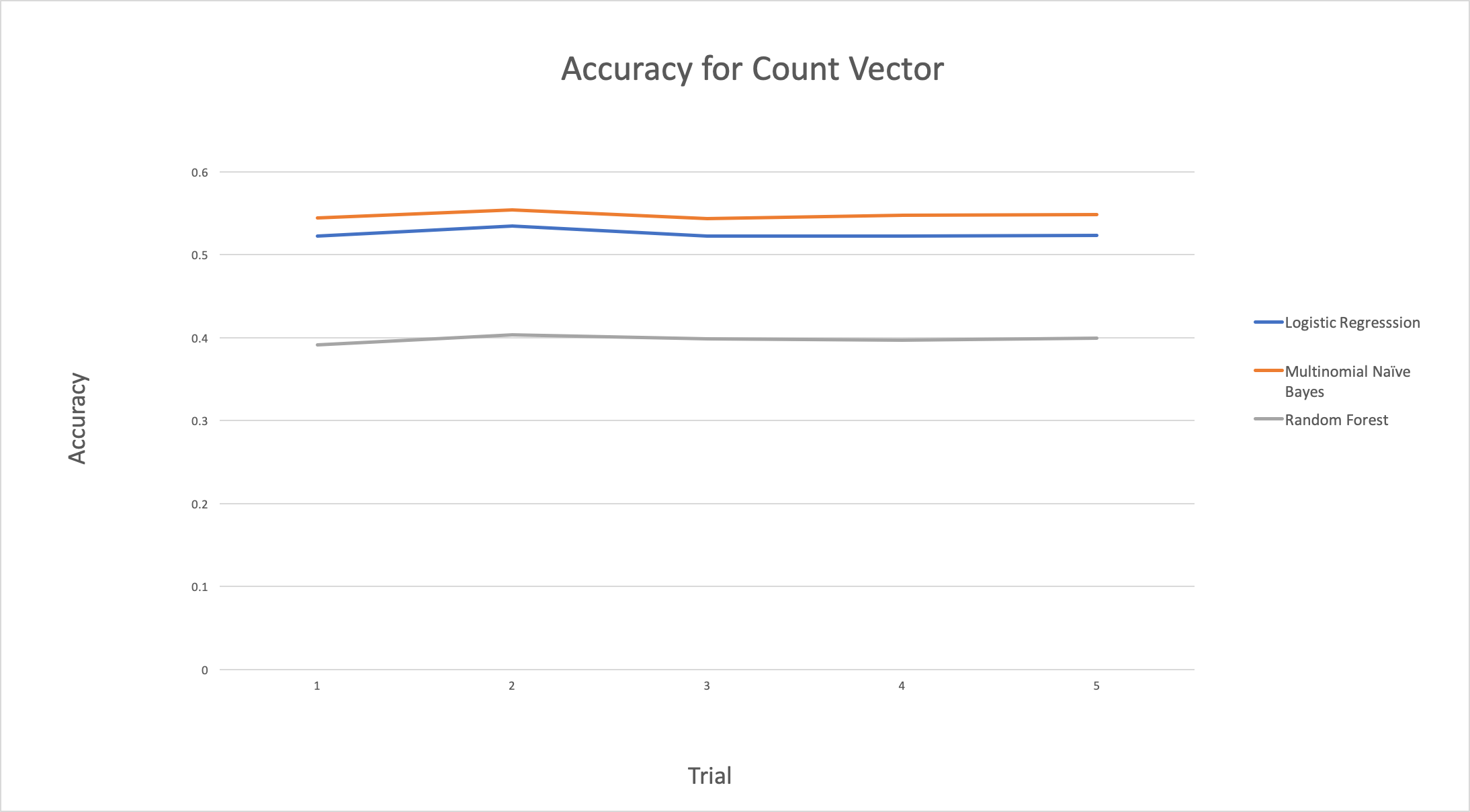
**Baseline Accuracies for Sci-Kit Learn Models**

|  |  |  |
| --- | --- | --- |
|  | TF-IDF | Count Vector |
| Logistic Regression | 54.80% | 52.53% |
| Mulitnomial Naïve Bayes | 56.18% | 54.77% |
| Random Forest | 39.95% | 39.80% |

****

*Accuracies for TFIDF Vectorizer from Sci-Kit Learn over 5 trials. This graph illustrates*

*how Logistic Regression and Multinomial NB outperformed Random Forest every trial.*

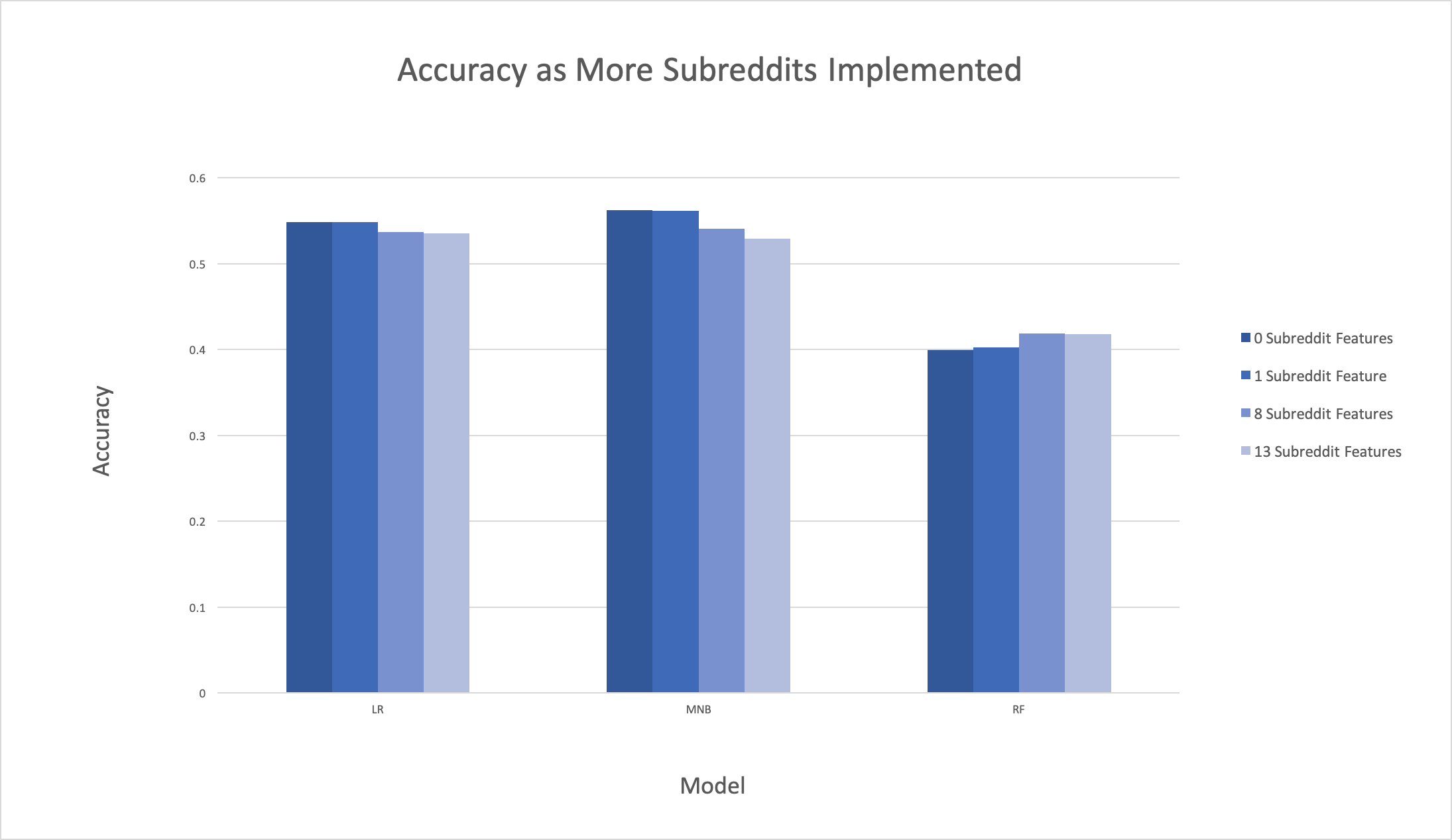
****

*Accuracies for Count Vector Vectorizer from Sci-Kit Learn over 5 trials. This graph illustrates*

*how Logistic Regression and Multinomial NB outperformed Random Forest every trial.*

As our data illustrates, Multinomial Naïve Bayes performed the best and Random Forest performed the worst on both feature matrices. The TF-IDF feature matrix also had better accuracy for all three Sci-Kit Learn models.

As we implemented our keyword vocabularies, we measured the accuracy trend whilst we added vocabularies for more subreddits:



*Accuracies for each classification model when 0, 1, 8, and 13 subreddit specific vocabulary features are*

*implemented. Logistic Regression and Multinomial Naïve Bayes decreased as we implemented more features,*

*while Random Forest increased.*

Our implementation of Bernoulli Naïve Bayes achieved an accuracy of 57%.

**Discussion and Conclusion:**

Our attempts at designing features for our Sci-Kit Learn models did not yield relevant increases in accuracy, which meant we were not able to increase our model accuracies by much. We hoped that by using features such as average sentence length, comment length, word count with stemming, and links, our model would be improve, however these did not earn us a significant boost in our predictions.

Naive Bayes models predict the class for a sample by computing P(Class k | sample x) = P(Class k) \* P(sample x | Class k) / P(sample x), or in words posterior = prior \* likelihood / evidence, but the exact implementation of these calculations may vary slightly from model to model. In this design, Sci-Kit Learn is used to preprocess the data: Vocabulary for the entire dataset is obtained, and each document is represented as a binary vector of equal length. To avoid unbalanced classes, the dataset is shuffled, and to avoid overfitting k-fold cross validation is applied, with k=5.

In each partition, the documents are grouped into classes. Prior probabilities for each class are stored as log-odds to save computation later. From the probability of each word appearing in a particular class, two vectors are precomputed, log\_p, representing log P(word | class), and log\_np, representing log 1 - P(word | class). As the output of the model consists only of predicted classes, and probability estimates are not required, the evidence term is not computed at all. The metric selected to evaluate the model is accuracy: For each test document, the class with the maximum posterior probability is selected and compared against the true value. While the model was less successful Multinomial Naive Bayes, which could make use of numeric tf-idf values, it performed quite well, with a typical accuracy of 49%.

-

When datasets are taken from the Web, analyzing the links embedded in documents can provide

valuable context. For instance, posts in a subreddit will often contain links to other posts in that same subreddit, to similar subreddits, and to relevant external sites. Certain comments in the dataset were submitted by 'bots', which reliably linked back to the subreddit in which they were posted (e.g. 'Your post was removed because it was really bad. Message the mods of /r/canada to complain.'). In practice, this feature was of limited value, as only about 10% of posts contained links.

The comments in the dataset were encoded in the Markdown format, which permits both semantic (e.g. headings, links) and non-semantic (e.g. bold, italic) formatting to be applied. In text classification it is common to strip formatting from text before performing other processing, to allow for words to be cleanly tokenized. Features could also be constructed which consider the type of formatting used in a particuplar comment. Interestingly, stripping formatting from comments slightly reduced the accuracy of the bag-of-words models tested. Perhaps Markdown syntax is sufficiently lightweight such that its presence provides additional information without introducing large amounts of extraneous tokens, as would be expected for HTML.

**Statements of Contribution:**

Nicholas Paun wrote the code for Bernoulli Naïve Bayes, the Pipeline construction, the link scraping feature, the word-to-vector feature and model selection. He also contributed to the write up.

Samuel Nordale wrote the code for the average sentence length feature, the comment length feature, the subreddit vocabulary features, and contributed to the Pipeline and GridSearch construction. He also performed inter-model analysis, created visuals, and contributed to the write-up

Hafsa Khalid wrote code for word count with stemming feature and model selection and made contributions to the write-up