Sci-Fi? Sci-Why?

Anish Shah and Jason Tran

**Introduction:**

Ever since the time man has made ink touch paper, the ideas and concepts hidden in the writer's head has always been a representation of the time in which they have written their words. Like spoken language, as the times change so do the words we use. No longer do we speak in Shakespearean English nor does the vernacular of that time match the slang of our age. Even looking at words from now a decade ago to now, there is an apparent and drastic change in the words we use and write. This concept is applied in any novel, any genre, and any form of communication. To get an overarching view of this concept and to see whether this assertion is correct, the isolation of one certain genre was necessary. Unlike abstract concepts like romance, or a more subjective bound genre like action or adventure it was best to look at the concepts that are bound by the time at which the writer wrote. In that sense the only real genres that did not rely solely on imagination nor creativity would be the ones that would need to have a basis in academics, a field which only reflect the degree of what was taught and available to the public at the time. The genre that matched these criteria turned out to be Science Fiction, a coveted genre indeed.

Upon analysis of the genre it was found that there were four distinct eras which we would be used to determine whether the aforementioned reasoning would indeed lead to distinct analysis of change of literature over time. The four eras are split into:

- The Golden Age (1940s – 1960s)
- New Wave (1960s – 1980s)
- Cyberpunk (1980s – 1990s)
- Contemporary (1990s – Now)

Each era tended to focus on many different Science Fiction topics that ranged from deep space to the deep web and as such the writing of each era should reflect these different topics by prioritizing the use of certain words over others. With that in mind the questions that wanted to be answered were "What era would a belong to given the language of the book?". As such to answer this question the use of EDA was needed to be able to analyze the word choice of each of these eras and after having a comprehensive understanding of the language distribution a complementary predictor could be made to directly address the issue of whether the words contained in a book can directly be used to predict the time period of said book (in this case specifically isolating the genre of Science Fiction).

**Approach:**

First and foremost, for this project before even considering analyzing books, books had to be collected. At first the idea was to collect the books from the most available medium which was pdf/e-book style. However, upon testing of Python packages like PDFMiner and Slate this form of file extension turned out to be extremely inefficient as it not only relied on packages not already inherently built into the Jupyter Anaconda tool but also returned the text of the book in an undesirable form. For instance, with the tool Slate, the function would return the content of the pdf page by page with each page being an index in a list and each word being an index of the page list. This tool, while giving a somewhat usable albeit tedious form, also retained all the formatting markers appended to the words. An example of this would be if a PDF contained:

*"Hello*
*World"*

The Slate tool would not return [‘Hello’, ‘World’] but instead would return [‘Hello\n’, ‘World’]. As it can be seen the newline character was explicitly retained and as such changes the stored word fundamentally. This is a huge issue when analyzing the books and creating the predictor as the word ‘Hello\n’ would not be treated the same as the word ‘Hello’ in the algorithm or when summing the count of each word occurrence in the book. Using regular expressions to remove these formatting characters was found to result in a lot of possible errors. To reduce all these unnecessary errors, the .txt file extension was used instead of the .pdf one as .txt files do not have formatting characters. This makes the parsing of the words of the file directly as how it appeared in the book separated with a whitespace character as the distinguisher between words.

After deciding to use .txt files as the medium at which the book would be stored, so then books representative of each era of science fiction was selected. A book for an era would be selected only if reliable resources gave indication that said novel was a pivotal book of an era that helped define said era. The number of books chosen to be used was 12 books in which there were 3 books per era where 2 books were treated as training data and 1 book treated as testing data. As such the 12 books selected are listed in the table below:

| **Golden Age (1940s-1960s)** | **New Wave (1960s-1980s)** | **Cyberpunk (1980s-1990s)** | **Contemporary (1990s-Now)** |
|---|---|---|---|
| The Martian Chronicles (1950) | Do Androids Dream of Electric Sleep (1968) | Metrophage (1988) | Down and Out in the Magic Kingdom (2003) |
| IRobot (1950) | Dune (1965) | Neuromancer (1984) | Infinite Jest (1996) |
| Triplanetary (1934,1948) | Slaughterhouse Five (1969) | Snow Crash (1992) | The Martian (2011) |

With all 12 books collected in the proper file format, only then was the analysis of the books able to take place. The first step that needed to take place was the rote separation of each word in the novel into a list to be analyzed further by the computer. To do that each line of the book file was read line by line where the words in each line were tokenized (stripped of all extraneous symbols, including the separation of contractions into their own individual word), made lowercase, and then appended to the end of a list. At this point a dictionary of each word and its count can be made by iterating through the list effectively reading the book word by word. Stop words from the StopWords package was implemented to prevent certain words from being counted as it was already understood to only cause issues of overtraining in the supervised learning algorithm or giving useless extraneous information. An example of this error is emphasized in the word “the” whose count would only overfit a decision tree.

Instead of using a simple dictionary, the Natural Language Toolkit package was implemented. This package contains many tools that can be used to process written language. As such their built-in Frequency Distribution object was used as this object could be used to do other types of analysis on the books. At this point the unique words of the novel and their count are now stored within a dictionary. This dictionary was then stored into a CSV file removing the need for the code needing to run each time to get the words and count in a

dictionary and more easily reading it from a CSV already containing that information.

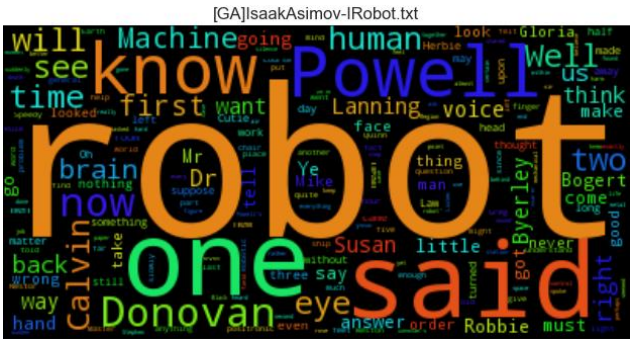
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | jonny | said | conover | back | like |
| 2 | 1540 | 689 | 347 | 329 | 325 |

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | case | said | like | molly | one |
| 2 | 845 | 664 | 318 | 279 | 264 |

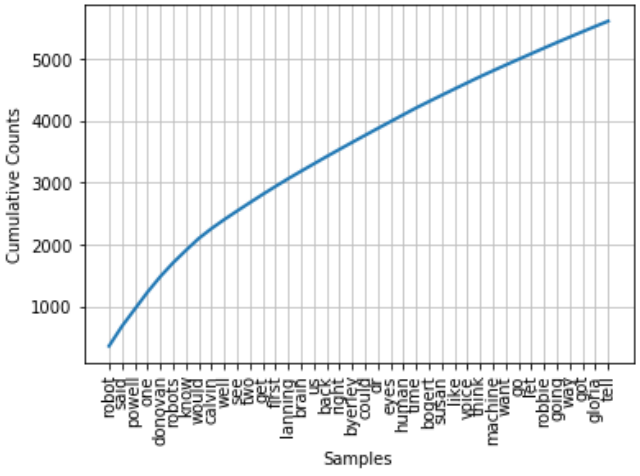
| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | jonny | said | conover | back | like |
| 2 | 1540 | 689 | 347 | 329 | 325 |
| 3 | 0 | 664 | 0 | 224 | 318 |
| 4 | 0 | 62 | 0 | 433 | 763 |
| 5 | 0 | 446 | 0 | 154 | 251 |
| 6 | 0 | 317 | 0 | 118 | 101 |
| 7 | 0 | 18 | 0 | 107 | 77 |
| 8 | 0 | 351 | 0 | 151 | 122 |

With CSVs made for each book, the only need thing that needed to occur was the adjoining of all these CSVs into one large CSV for the same aforementioned reason. As such each CSV was read and a list of the words was stored and only added onto if the word was not already in there. Then the CSV was read again to get the count of each word if in there and zero if not. This CSV then contains every unique word from each book with the removal with selected stop words and their count in each book. The columns would be the word and each row would be a book. All the eras have been clustered together in this CSV.

**Results:**



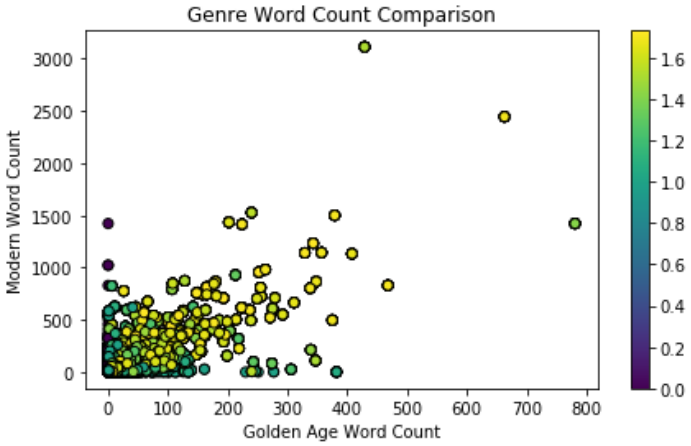

One of the focuses of this project was to gain insight on the data contained in each of the books and gain experience with different packages that help give that insight. As such the WordCloud package was implemented to give a visual weighted list of each word and its frequency in the book by increasing the size of the word in the image proportionally. As such this gives an image that corresponds to the CSV of a book as outlined in Approach. This image also gives insight into what words seem to have the most effect on the content of the book as the frequency of a word in a novel can indicate the importance of a word to the content of the story. This is shown in the sample word cloud for the novel IRobot of the Golden Age era of science fiction. As it can be seen there is an overwhelmingly higher amount of the usage of the word “robot” in the novel which is highly indicative of the subject matter of the book. However, an example of a word that may hold no importance to the subject matter of a book is the word Donovan as it is a name and names really have no indication as the matter of a book (unless the name was extremely unique and said uniqueness made it a strong indicator).




With this in mind, the Natural Language Toolkit was used to gain access to the cumulative frequency distribution plot of a book’s top 40 most common books. This plot is used to show that after a certain number of words, the cumulative count curve tends to taper off indicating

that a word is used severely less frequently than the other words whose slope at that point is much higher (meaning the word was used a lot more that its right-hand neighbor). A book is generally defined to have 80,000 non-unique words and as such by using this cumulative count curve and the concept of tapering off mentioned earlier, the percentage a book is composed of n most common words can be reasonably estimated/calculated.

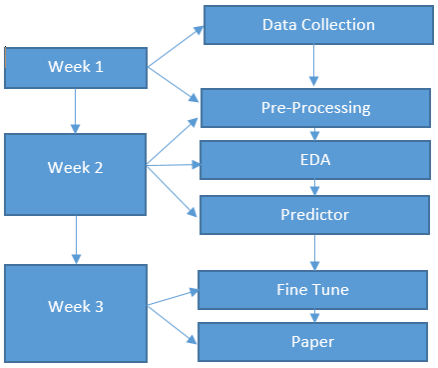
At this point the books were converted into CSVs and then merged to create the cumulative total word list CSV. Using this CSV, a numpy array was made after removing the header row. Then at this point the books of the same Science Fiction era were grouped together for analysis to discover which words would be more distinct in one era over another. This was done by summing all the counts per word per era to create a n word by 1 row vector per each era. Then the vector was transformed into a Term Frequency Inverse Document Frequency to allow for proper coloration as it normalizes the data into a way that allows for easier viewing of which word belongs to what era. Thus, when placed into matlibplot's scatter function a plot that compares the counts of a word in each of two eras is made along with the degree of comparison of the word to each era (with the more central combined color meaning that the word was indistinguishable between eras and any value above or below that meant said word was more indicative of one era over another). As such the words with the largest degree of change from the central green point were the words that signify the biggest indicators of an era over another with green meaning there was no helping factor as to the differentiation of eras. Yellow meant that the word was a more Golden Age book and blue meaning it was a more Contemporary book. This insight allows for better training of an accurate supervised learning model.




```
Nearest Neighbors accuracy: 0.25
Linear SVM accuracy: 0.25
RBF SVM accuracy: 0.25
Decision Tree accuracy: 0.25
Random Forest accuracy: 0.5
AdaBoost accuracy: 0.25
Neural Network accuracy: 0.5
Linear Regression accuracy: 0.354452927209
logistic regression accuracy: 0.25
```

To create our supervised learning algorithm, we simply iterated through many supervised learning models after normalizing the dataset. After initial training, the hyperparameters of each of the different models were tweaked with until it reached a fairly reasonable degree of consistency. It can be seen that the majority of the models yielded accuracies of 25% with some even reaching 50% accuracy. The best accuracies were found to be mostly in the Decision Tree model, the Neural Network model, or the Logistic Regression model. In the Decision Tree, it was seen that the default depth of 10 was not enough to provide a good predictor so that was continually increased until it reached a point where it decreased accuracy yet again (as in overfitting). In the Neural Network, the weight optimization hyperparameter was changed to stochastic gradient descent. Furthermore, the learning rate of the model was increased by a factor of 10 in comparison to the default to try to mitigate the issues training with smaller sample sizes. Overall, this variable degree of accuracy can be attributed to our relatively small number of samples.

**Conclusion:**




From the timeline presented on the side, in which the size of the week boxes represents the relative time on each task it can be seen that the process of classifying Science Fiction books into a certain era can be broken into distinct segments that flow into each other. These segments being Data Collection, outlined in approach, which is the process undertook to actually get the files of each book and making sure they were in a usable format without formatting errors; Pre-processing which was the conversion of said books into CSVs of lowercase tokenized words with the removal of certain stop words and their count to remove the need to explicitly re-store the contents of a book into a list upon shut down of the Jupyter model along with the adjoining of all the book's CSV's into a comprehensive list of all possible words and their appearance/count in a book. This means each word was a column in the CSV and each row being a Science Fiction book. At this point the CSV could be used to create a variety of different plots as seen in the EDA segment of the timeline. The different plots used were:
1.) a word cloud to give a visual representation of what we expect to generally see from the conversion of a book into a CSV of a word and its count where a bigger word indicated a higher frequency;
2.) a cumulative count curve from the NLTK package that gave the compounding makeup of a book after plotting the first 40 most common books. As such it shows that after a certain point the curve tapers off showing that at that point the word in question does not have a drastic effect on the composition of the book; and a
3.) a matlibplot scatter plot of the sum of the count of words of two eras plotted against each other with coloration given by the TFI normalization of each vector. The darker/lighter the color a point was, the more closely correlated that word was to one era over another. Similarly, the more mixed/dull the color was, the less indicative that word was in terms of era classification.
It was at this point that the understanding of the data was used to train predictors for classifying a Science Fiction book into a certain era. The best classifiers were the Decision Tree with hyperparameter of depth changed to 80, a Neural Network with hyperparameters of weight optimization changed to gradient descent and learning rate increased by a factor of 10 compared to the default. Generally, the accuracy only ever amount to 25%

Overall the outcome of this data science project is lackluster at best and that can be accredited to the small sample size of 12 books. Further growth of this project can include splitting each book into distinct chapters that instead can be used as the samples for era prediction, thus not causing an unreasonable growth to the size of data but increasing the sample size thus giving us the ability to have more training and testing data.

**References:**


1.) http://www.nltk.org/book/
2.) https://matplotlib.org/contents.html
3.) http://scikit-learn.org/stable/documentation.html
4.) https://www.kaggle.com/apapiu/predictions-in-the-republican-primary
5.) https://github.com/timClicks/slate