*Final Project: Predicting a recommended review based on natural language.*

*Alice Brenengen*

**1. What is your central research question? How will your work contribute to the problem that you have defined?**

The goal of this project was to create a model to predict whether or not a reviewer would recommend a game based on the language used in their review. My data originally consisted of over 400,000 reviews of different video games from Steam but was parsed down due to processing time. My job was to import and clean data as well as create the model itself.

**2. Explain in short the way you handled the solution. You may want to include something you struggled with and managed to solve.**

Initially, when I created my model I wanted to use the entire dataset. However, creating a model with over 400,000 observations was not possible using my computer so I had to find a way to consolidate the data. I started by using filter() to select reviews with specific titles (essentially using reviews from a single game). From there I went even further: I found the number of hours of gameplay in an average video game (20 hours) and how long you would have to play the game in order to construct a fully formed opinion (1/4 of the game). This way I could narrow down the reviews by the number of hours the review had played the game and take out observations where the reviewer had not played the game long enough to be able to write an adequate review (20 * 1/4 = 5 hours [or less]).

Next, I did some cleaning and encoding to make the data easier for R to work with. I began by determining if a review was funny/not funny and helpful/not helpful. To do this, I found the average number of funny/helpful votes and then used ifelse() to replace the number of funny/helpful votes with a 1 if the number of votes were higher than the average or 0 if not.

Then, I encoded using factor(). This step was not super necessary since I ended up using natural language processing but at the time I wasn't sure what I wanted to do with the data and it's always good practice as well as prep in case I want to use this data for another project in the future. Lastly, I encoded the recommendations column by first making the data type a factor with is.factor() and then encoding the same way as before.

After the data was cleaned and ready, it was time to create the Corpus. This was the most difficult part of the process due to the size of the dataset. I struggled a lot with finding the right sparsity so that my computer could run the code properly, most of it was trial and error. Eventually, the combination of consolidation, sparsity dropping, and Shivani help, it worked. Once I was able to create a matrix of all of the words used in every review and which reviews used them, I added in the recommendation column from the original dataset. This way we could split the data into test sets and create classifiers from them. The programs could then correlate the words used with whether or not the reviewer recommended the game and create a model that could be applied to any review.

**3. Display and explain your results**.

I used 3 different modeling methods; Logistic Modeling, Support Vector Machine (SVM), and K-Nearest Neighbors (kNN). Given the smaller dataset the accuracy of the models were not amazing but could possibly be made better by using more observations. The model with the highest accuracy was the SVM model at 77.3%. This means that the model has a ~78% chance of correctly predicting whether or not the reviewer would recommend the game based on the words used in the review itself.