Unsupervised/NLP Project Write-Up

By: Colin Gallagher

Abstract:

A recommendation system was built using natural language processing and unsupervised learning techniques to recommend games to consumers on the Steam online game marketplace. Data was scraped from the Steam store website and from games' store webpages. The information that was recorded was the written descriptions for each game as text data. Using Python, the data was pulled in, cleaned, and several topic modeling methods were applied to create the final recommendation system. Clients that would benefit from using the recommendation system would be Steam as a seller and their consumers. Their consumers would be recommended more games that they are interested in and buy more, thus Steam selling more. Both parties benefit.

Design:

The purpose of this project is create a recommendation system using natural language processing and unsupervised learning methods. The problem that this recommendation system solves is a way for Steam to recommend games to their customers by using only the description of the games given to them by the developers or creators of the games. There is no need for having any experience with each game, the data for the content is all from the description supplied to the store. A brand new game listed can immediately become part of the system. The only drawback being: how well does the description carry over to the actual gameplay experience?

Data:

The raw data that was used for this project was scraped from the Steam store website. Each game has its own webpage with all of its relevant information. The main data of interest was the written description given for each game. A total of 1,915 games' webpages were scraped. Each page had information such as title, developer, publisher, genres, tags, and most importantly the description. Every page represents a single game being a separate row of data, and each column would be the information previously stated. However, the only columns used for this

project ended up being Title and Description.

Algorithms:

To scrape the data, first a list of links of the top 2,000 games was produced. Using this list of links the data that needed to be scraped was scraped: most importantly title and description of each game. There were games that were region blocked in the United States or required a login to view the page, so not all 2,000 games were included in the final data set, just 1,915. A corpus was made with each descriptition being a document. To start, a document term matrix was created using CountVectorizer and TF-IDF Vectorizer. Using these two document term matrices, topic modeling was done to assign topics to each document/game. Latent Semantic Analysis, Non-Negative Matrix Factorization, and Latent Dirichlet Allocation were all used for the topic modeling process. An intereative process was then started to help with the corpus cleaning. After each topic was made, the top words for each topic were analyzed and very general popular terms were continued to be removed until the final results were acceptable. These terms were added to the stop word list. Once the junk words, links, and numbers were removed, the number of topics was adjusted to a liking that made the most sense. The number of topics that was found to best clearly define and organize games into respective topics was 10. To pick the topic model between LSA, NMF, and LDA a similar logic was used: which model best separates the topics into clearly defined areas with little overlap? The topic modeling that was found to achieve this best was NMF with the TF-IDF matrix. The 10 topics could be renamed as: Hero Adventure, Real Time Strategy, Cars/Racing, Survival, Action Story-Based, Simulation/Tycoon, Outerspace/Futuristic, Comic/Movie Heroes, Sports Simulation/Management, Multiplayer Shooter. Now that a final document-topic matrix was produced, this is what would be used as the data coordinates for the recommendation system. To avoid the curse of dimensionality the 10 column matrix was reduced to three dimensions using principal component analysis for the distance calculations. The recommendation system is very simple. Games that are recommended are ones that are closest together. A function was defined for the user to input a game title and number of similar games to return. It was found that based off intuition/experience the recommender works well for most games, but there are a few instances were game recommendations are questionable.

Tools:

Tools used for building the recommendation system:

- **Python**: Coding language used to perform actions on the data with its various libraries.
- **Pandas**: Python library used to clean and place data into organized data frames.
- **BeautifulSoup**: Python library used to scrape and parse through HTML to get the data.

- **re**: Python library used to make regular expressions for cleaning the documents in the corpus.
- **Sklearn**: Python library used to vectorize documents in the corpus and do topic modeling on the document term matrices.
- **Matplotlib**: Python library used to visualize the data points.

Communication:

A five minute slide presentation will be given to explain the recommender, how it was created, the results, conclusions, any future work that can be done to improve the project, and answer any questions.