Vectorized Representation: Movie Reviews Data

Dhaval Patel

Northwestern University

Abstract

The objective of this project is to enhance a corpus-wide vocabulary and derive several insights from 200 movie reviews using Natural Language Programming (NLP) approaches such as data wrangling with document vectorization techniques such as Tf-idf, Word2Vec, and Doc2Vec. Various data wrangling techniques and approaches were used to study the documents and get a knowledge of each methodology. This will enable us to create a corpus vocabulary for clustering and classification in the following stage.

Introduction

Researchers in the field of Natural Language Processing (NLP) have created techniques for analyzing the content and linguistic structure of enormous corpora of texts (literally Latin for body). Frequency counts of items in a text are provided through corpus studies. This enables researchers to look for and identify significant words and phrases as well as study their concordances, words that appear around them, collocation, identify, and extract correlated keywords. This makes it possible to look at how words are used in context.

Methodologies for doing research incorporate both quantitative and qualitative methods, with subjective assertion drawn from quantitative data resulting in the development of advanced natural language processing systems.

Literature Review

Methods

Data Preparation, Exploration, Visualization Process

For this project, 200 reviews of the following 20 movies were chosen from IMDB. Rotten Tomatoes, and other movie rating articles in order to efficiently construct a text corpus: ['Angel Has Fallen' 'Inception' 'No Time To Die' 'Taken' 'No Time To Die' 'Taxi' 'Despicable Me 3' 'Dirty Grandpa' 'Grown Ups' 'Legally Blonde' 'Lost City' 'Drag me to Hell' 'Fresh' 'It Chapter Two' 'The Toxic Avenger' 'Us' 'Batman' 'Everything Everywhere All at Once' 'Minority Report' 'Oblivion' 'Pitch Black']

The process of extracting phrases, names, locations, and concepts from each review was performed manually. The following are a few key terms that should be familiar to everyone interested in learning more about the movie ‘Us’: Jordon Peele, family, vacation, boat, shadow, escape, and beach. Using Termine and FiveFilters analysis to perform automated term extraction,

counting, and ranking for each review, we found that the most frequently used terms had to do with providing background information about the film, such as the names of the directors, the actors, how the film related to the genre, and some terms leading up to providing story context. Based on last week's assessments on the movie Us, these are the keywords that stood out as most important: candidate_terms =

['horror', 'wilsons', 'family', 'adelaide', 'get out', 'peele', 'nyong', 'vacation', 'doppleganger', 'film', 'park', 'california' , 'monsters', 'beach', 'scary', 'escape', 'shadow', 'us', 'boat', 'scissors']

Gensim, a free open-source Python library for topic modeling, document indexing, and similarity retrieval utilizing large corpora. This library is intended to handle raw, unstructured digital texts ("plain text") employing unsupervised machine learning techniques. Figure 1 depicts all of the other imported packages that were used.

```
import pandas as pd
import os
import numpy as np
import re
import string
import seaborn as sns
import matplotlib.pyplot as plt
import nltk
import random
from dataclasses import dataclass

from nltk.stem.wordnet import WordNetLemmatizer
from nltk.stem import PorterStemmer

import gensim
from gensim.models import Word2Vec
from gensim.models.doc2vec import Doc2Vec, TaggedDocument

from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.manifold import TSNE

import scipy.cluster.hierarchy

from IPython.display import display, HTML

from typing import List, Callable, Dict
```

Figure 1. Packages

The corpus has 200 rows representing the movies reviews and 9 columns representing DSI_Title, Submission File Name, Student Name, Genre of Movie, Review Type (pos or neg), Movie Title, Text, Descriptor, and Doc_ID. The number of reviews is 50 which is balanced across the four genres of movies: Action, Comedy, Horror, and Sci-fi.

Researching Design and Modeling Methods:

To obtain understanding of the corpus data, both qualitative and quantitative methods were applied. The terms used to identify or define a movie were evidently chosen by students as part of the qualitative method, with each student examining 10 reviews of a movie. We chose terms that were frequently used in movie reviews. The following data wrangling techniques were applied:

Method 1: normalization and tokenization

Method 2: normalization, tokenization, lemmatization, and stop words

Method 3: normalization, tokenization, lemmatization, stop words, and removal non-alphabetic

Firstly, before we diving into each methods, we calculated cosine similarity of all 3 data wrangling methods the cosine between two vectors by the following formula (Lane, Howard, and Hapke, 2019): $cos\theta = \frac{A.B}{|A||B|}$ where A and B are different vectors. The closer the cosine similarity measure is to 1, means the more similar are the two vectors. The cosine similarities can be analyzed through heat maps for each data wrangling methods how each movie review to similar to another movie review in the class corpus, based on their tf-idf vectorization score.

Explanation: Method 1

Normalization and tokenization are part of the first method because data contains noise, we put the text in a standard format, a process known as normalization, to remove terms that provided no vital information about the text. To clean up the text, the following procedures were used: remove punctuation, lower case, tags, and special characters and digits. Tokenization was accomplished using the NLTK Python package. Tokenization is a form of document segmentation that divides a text into smaller bits, such as paragraphs, sentences, phrases, words, or terms. Each component is referred to as a token. As a result, these bits of information may be

counted as highly abstract and utilized to represent the document as a vector.

Explanation: Method 2 and 3

Lemmatization and stop words were two additional phases in the second method. Lemmatization is the process of removing a word of its linguistic ends and reverting it to its lemma, or dictionary-based form. Stop words: Occasionally, some exceedingly frequent words that may appear to be of little value in aiding the user in choosing documents that meet their requirements are eliminated from the vocabulary. The non-alphabetic tokens were also eliminated from the corpus using a further step in the third method.

Explanation: Term Frequency - Inverse Document Frequency

We computed the Term Frequency - Inverse Document Frequency, or TF-IDF, for each technique. The acronyms TF (Term Frequency) and IDF are combined to form the word TF-IDF (Inverse Document Frequency). A document's word frequency is the first factor considered by term frequency. The word frequency is then normalized because each text has a distinct length. When compared to the frequency of the same term on a smaller document, a word will appear more frequently on a larger document. It's generally calculated as follows:

$$TF(w) = \frac{Number\ of\ times\ the\ word\ w\ occurs\ in\ a\ document}{Total\ number\ of\ words\ in\ the\ document}$$

The primary objective of a search is to locate relevant documents matching the query. We need a mechanism to reduce the weight of phrases that appear too frequently and increase the weight of terms that occur less frequently. This is achieved through the term Inverse Document Frequency, which measures the significance of a term inside a document. It is essentially the number of corpus documents that include the term. This frequency is then inverted to ensure that less common but potentially relevant terms are given priority. In contrast to Term Frequency, Inverse Document Frequency increases the weights of relevant but less frequent terms while decreasing

the weights of frequent terms. Integrating Term Frequency with Inverse Document Frequency yields the TF-IDF model formula. $weight(w,d) = TF(w,d) \times IDF(w)$ To learn distributed representations (word embeddings, correspondingly, document embeddings) required in the following stages when applying neural network, word2vec and doc2vec visualization tools corresponding to the three data wrangling methods and different dimensions, 100, 200, and 300 dimensions, were conducted. Word2vec generates embedded words. Each distinct word is associated with a vector in the dimensional space. Word vectors are positioned in the vector space, with words that have similar contexts in the corpus being situated in location. Word2vec considered 100, 200, and 300 randomly selected words, the maximum distance between the current and predicted words (neighboring words) inside a phrase (in our instance, set to 3), min count (ignored all words with a lower overall frequency), and the embedding size. Doc2vec is a tool that generalizes the word2vec approach and allows documents to be represented as vectors. Regardless of a document's length, doc2vec objective is to provide a numeric representation of it.

Results, Analysis and Interpretation

TF-IDF Vectorization Analysis: We transformed the documents into a matrix of token counts using sklearn.feature extraction.text.CountVectorizer. The most frequently occurring words in the horror movie 'Us' were identified using the mean frequency indicator and the count vectorization technique. Figure 2.

	Us	All Horror	All Non-Horror
us	4.20	1.10	0.31
family	3.10	0.64	0.37
horror	2.70	1.76	0.08
film	1.90	2.34	2.59

The most frequent words in *Us* From the table, we noticed that the most words often used in the reviews: horror,

film, family, and us. Furthermore, these words also were shown to be present in “all horror” reviews but some terms were in some “all non-horror” reviews. The Term Frequency - Inverse Document Frequency (TF-IDF) approach is used to quantify terms in a collection of documents. A score is assigned to each word to indicate its significance in the corpus. The figure 3 below depicts the mean TF-IDF for the corpus for each previously reported data wrangling method:

tf-idf vectorization Class Corpus Results:

Data Wrangling Method 1	**Data Wrangling Method 2**	**Data Wrangling Method 3**
baseline = tokenization + lowercase	baseline + lemmatization + stopwards	baseline + lemmatization + stopwards + alpha only
Mean TF-IDF the 27.39 and 14.05 of 13.50 to 13.01 in 8.20 is 7.92 that 6.25 it 5.83 as 4.46 with 4.43 Vocabulary size: 12159	Mean TF-IDF film 3.65 movie 3.04 ha 2.56 wa 2.41 one 2.29 like 2.17 time 1.98 get 1.72 bond 1.70 character 1.54 Vocabulary size: 10862	Mean TF-IDF film 3.65 movie 3.04 ha 2.56 wa 2.41 one 2.29 like 2.17 time 1.98 get 1.72 bond 1.70 character 1.54 Vocabulary size: 10862

We discovered that when simply normalization and tokenization were used, words of low value were chosen: "to," "in," "it," and so on. While the other two techniques enhanced our search and reduced our vocabulary by about 2,000 words, the findings in the last two tables were remarkably similar. That is, following lemmatization and stop words in method 2, the extra step in method 3 of deleting non-alphabetic tokens had a lower impact on the corpus. Because "film" and "movie" are similar, including both in the word corpus is irrelevant to our research. Also, "like" is unclear since it might be a preposition indicating likeness or a verb indicating emotion.

Only "film" appears in the list above, with a TF-IDF score of 3.65 for the second technique 2 and 3.65 for the third method are similar not much changed, from the list believed to be important and extracted based on manual and automated extraction for the corpus-wide vocabulary. Based on these findings, the recommended class vocabulary was family (unique to the film under detailed examination). The following tf-idf score make sense since the words in the movie 'Us', family, and vacation could easily be highly correlated to different genre movies in class corpus such as ‘Grown Ups’ or ‘Dirty Grandpa’.

Data Wrangling Methods Heatmap Results Analysis

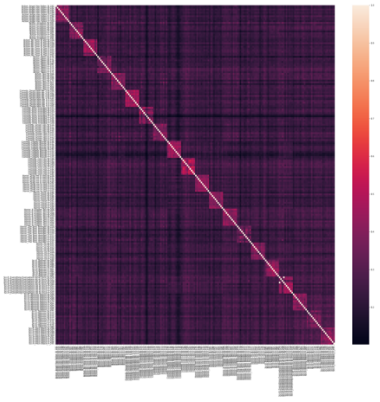
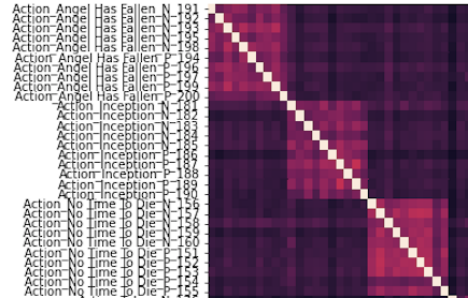


Figure 4 TF-IDF Heat Map Method 1

(Zoom In: Heat Map for Analysis)



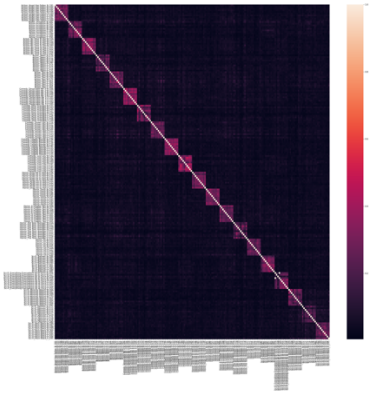


Figure 5 TF-IDF Heat Map Method 2

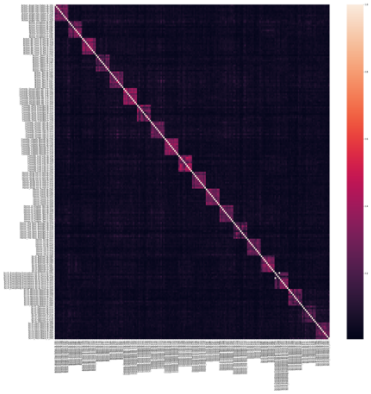
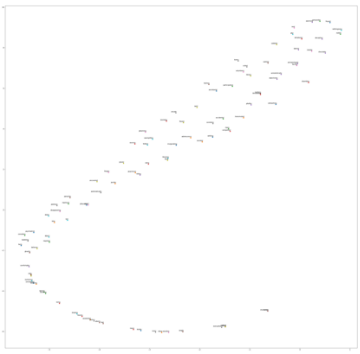
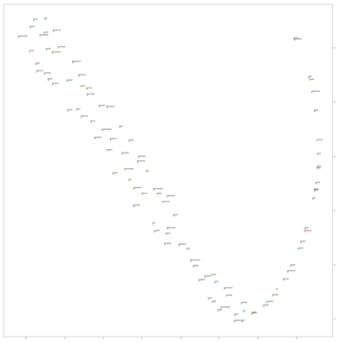
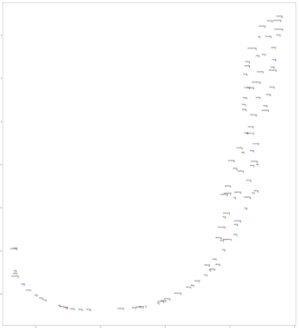


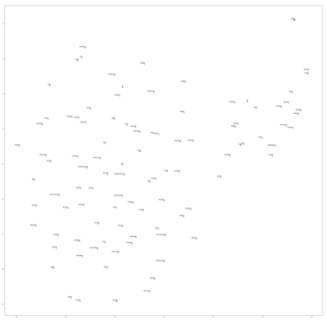
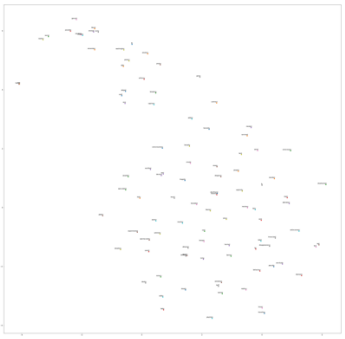
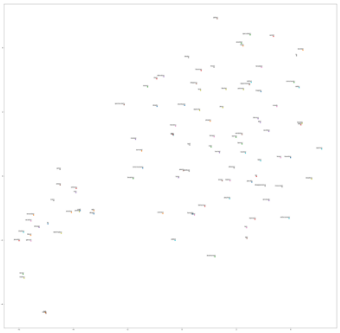
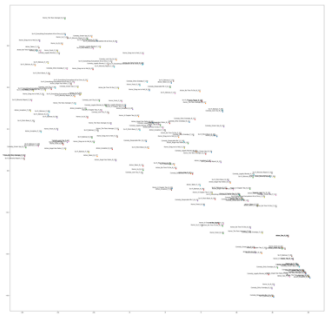
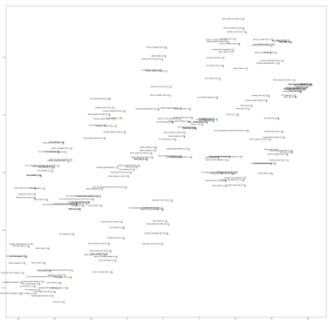
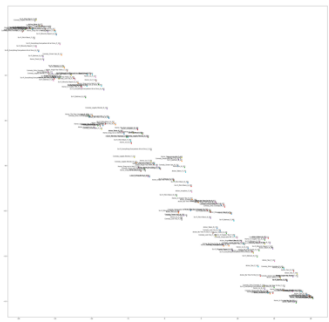
Figure 6 TF-IDF Heat Map Method 3

The TD-IDF heatmaps were generated using the cosine similarity of the reviews' 200x200 matrix. The cosine is 1 along the diagonal, indicating that two documents are identical. We also discovered that all 10 reviews of one movie in each genre contain similar words, since the cosine is frequently more than 0.6. Figures 4, 5, and 6 demonstrate a strong distinction between the 10

reviews of one movie but less among remaining genre movies reviews in the class corpus. Zooming in on the first heatmap, Mike Banning, Angel Has Fallen N191 is comparable (bright color, high cosine) to Mike Banning, Angel Has Fallen N192, but not to movie Inception N181 (dark color, low cosine) (fig.4). The first heatmap also reveals that the 200 reviews are rather similar because our vocabulary corpus contained numerous stop words and completely unrelated to a review terms (see the TD-IDF tables above), the initial heatmap revealed that the reviews are mostly similar. However, after employing more data wrangling techniques, the segmentation improved. The latter two heatmaps using data wrangling method 2 and 3 are darker, indicating a low cosine similarity. As a result, we determined that more complex data wrangling approaches aided in the better segmentation of reviews by movie title. Furthermore, we found little resemblance across genres, therefore genre grouping will be tough to generate.

Figure 7: Word2Vec t-SNE plots

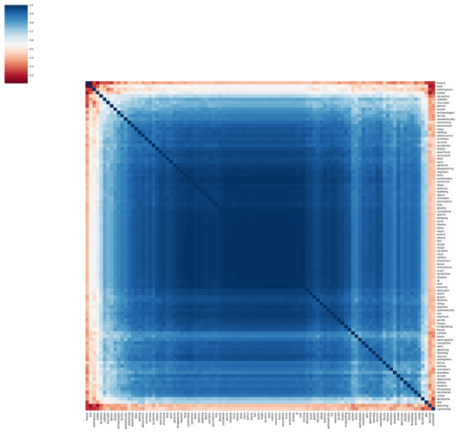
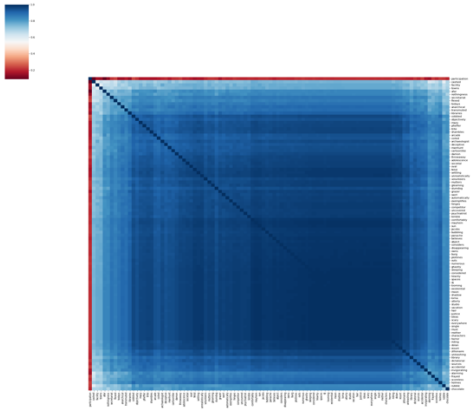
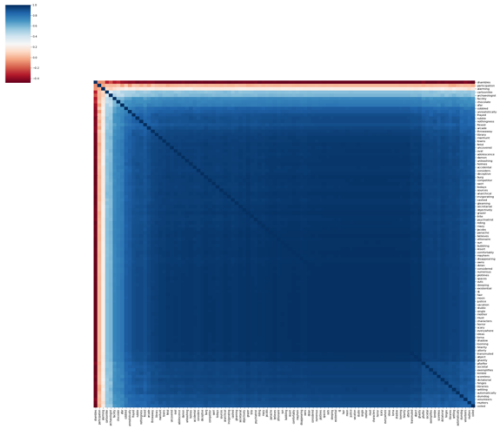
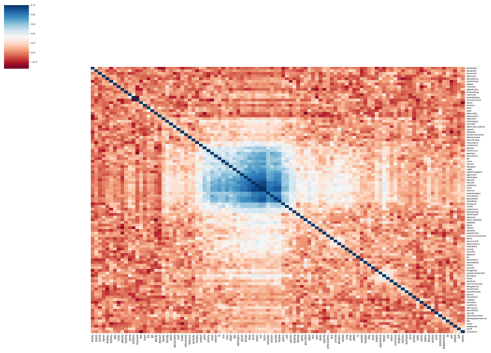
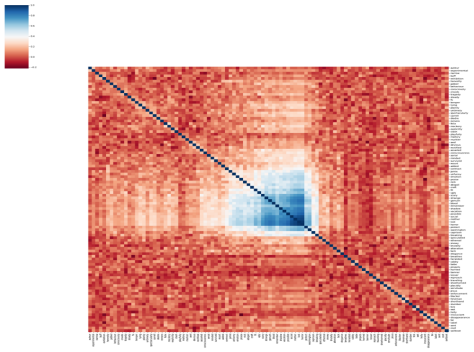
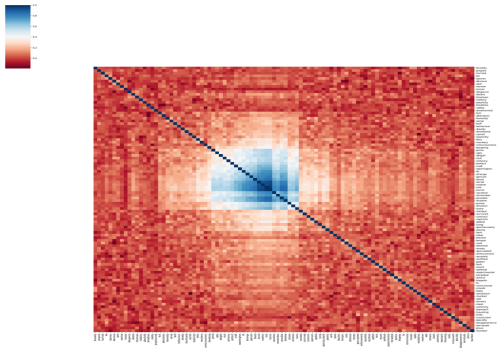
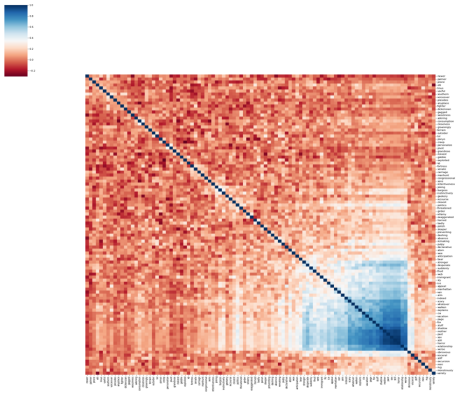
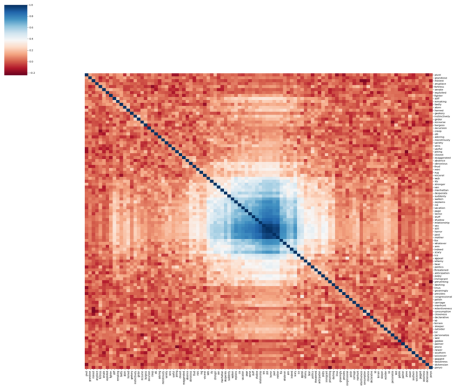
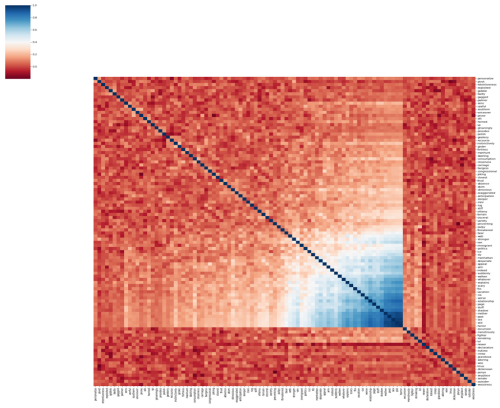
	Embedding Sizes		
Methods	**100**	**200**	**300**
Data Wrangling Method1: baseline = tokenization + lowercase			

Data Wrangling Method2: baseline + lemmatization + stopwards			
Data Wrangling Method3: baseline + lemmatization + stopwards + alpha only			

The figure 7 shows word2vec algorithm tsne plot analysis for three different data wrangling methods, and different embedding sizes of 100, 200, and 300. The tsne-plot is a 2-dimensional scatter plot where we are trying to visualize and analyze if these random 100 words plotted have clusters of similar words from between different movies reviews or genres based on word2vec algorithms. However, as the t-SNE charts for method 1 and 2 have interesting shapes but do not form visible clusters. But, in the data wrangling method 3 we do see some clustering being forms as the embedding size is increased to 300-dimensional vector space. For example, movie ‘Dirty Grandpa’, ‘Us’, and other movies are both clustered together in the chart t-SNE plot which make sense from analysis perspective since word ‘family’ and ‘vacation’ might correlated in both movies even though they are different genres. I would recommend data wrangled method 3

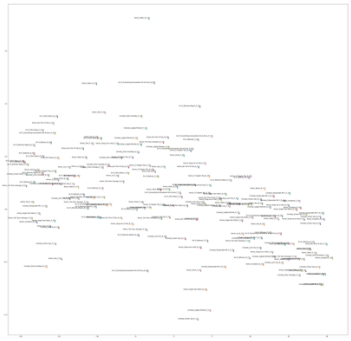
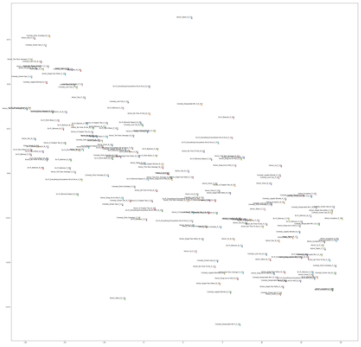
based on t-SNE plot analysis since they are visible formed clusters compared other methods.

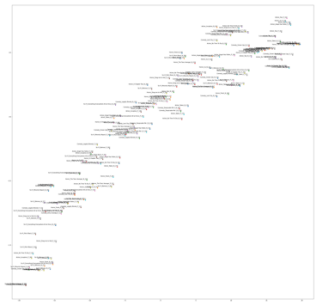
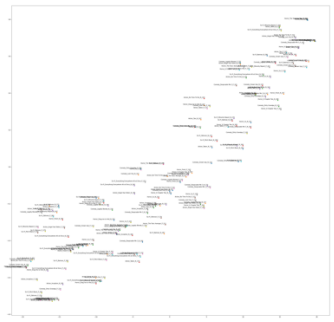
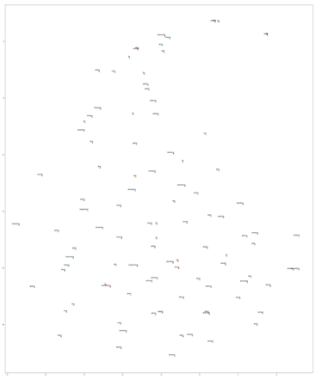
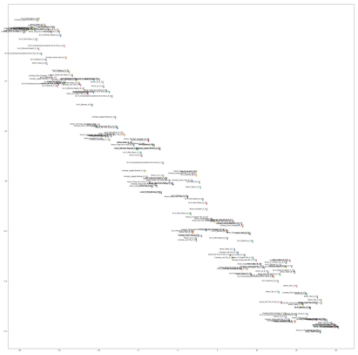
Figure 8: Word2Vec Chart HeatMap (All 9 Experiments)

BL_Word2Vec_100	BL_Word2Vec_200	BL_Word2Vec_300
		
M2_Word2Vec_100	**M2_Word2Vec_200**	**M2_Word2Vec_300**
		
M3_Word2Vec_100	**M3_Word2Vec_200**	**M3_Word2Vec_300**
		

The figure 8 shows word2vec algorithm heat map clustering analysis is based on cosine similarity for three different data wrangling methods on 100 words for all 9 experiments, and different embedding sizes of 100, 200, and 300. Based on the observation we can see its difficult to visibly analyze a well-defined cluster as most of the charts in Method 2 and Method 3 with embedding size 100, 200, and 300 shows colored light to heavy blue which coincides with cosine similarity of 1. Based on our analysis well-defined clustering map is the second data wrangling method and word embedding size of 100 which is M2_Word2Vec_100. When closely observed the chart, the cluster does have a lot of light to dark blue colored in the middle of chart showing similar words such as emotion, strange, social, mother, shadow, and vacation shows is closely related to different genre and movie in the same context. However, the word clustering doesn't appear to be consistent for Method 2 chart. As a result, word2vec clustering analysis doesn't appear to provide consistent clustering since corpus size is small. But for larger corpus size with classification techniques such as LSTMs, and GRU neural networks it can be useful.

Figure 9: Doc2Vec t-SNE plots

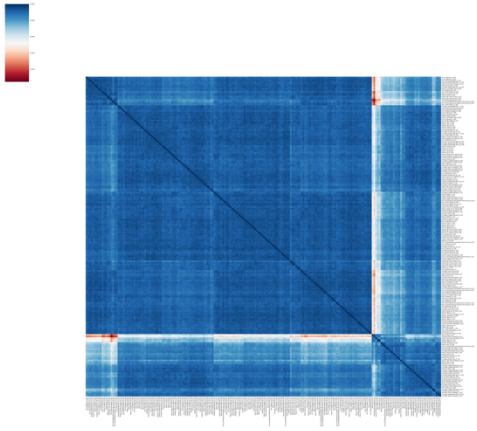
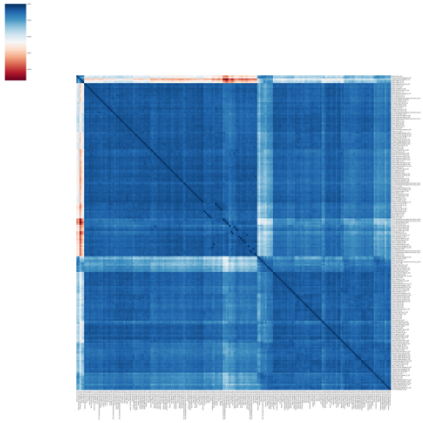
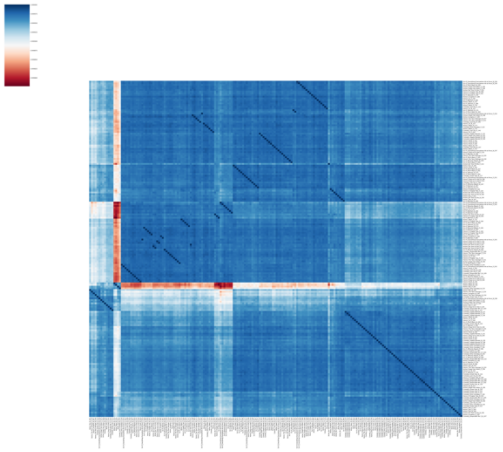
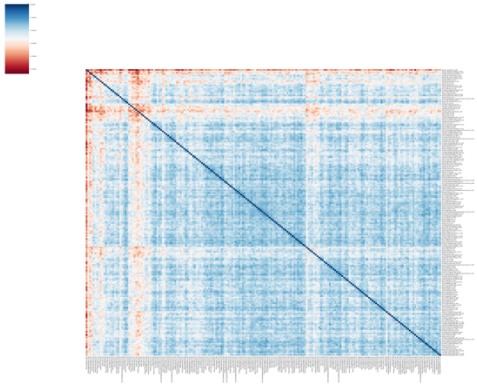
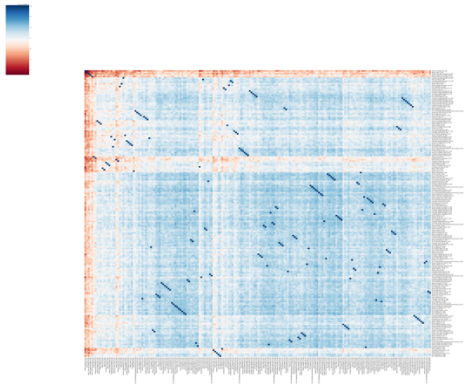
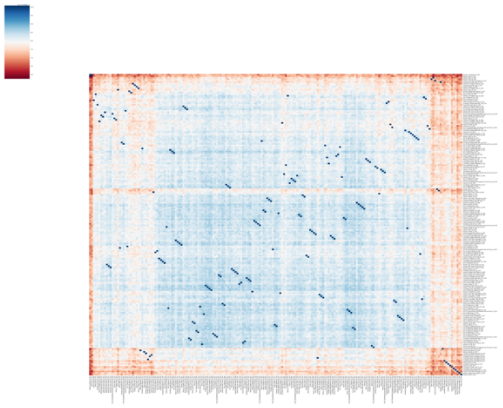
	Embedding Sizes		
Methods	**100**	**200**	**300**
Data Wrangling Method1: baseline = tokenization + lowercase			

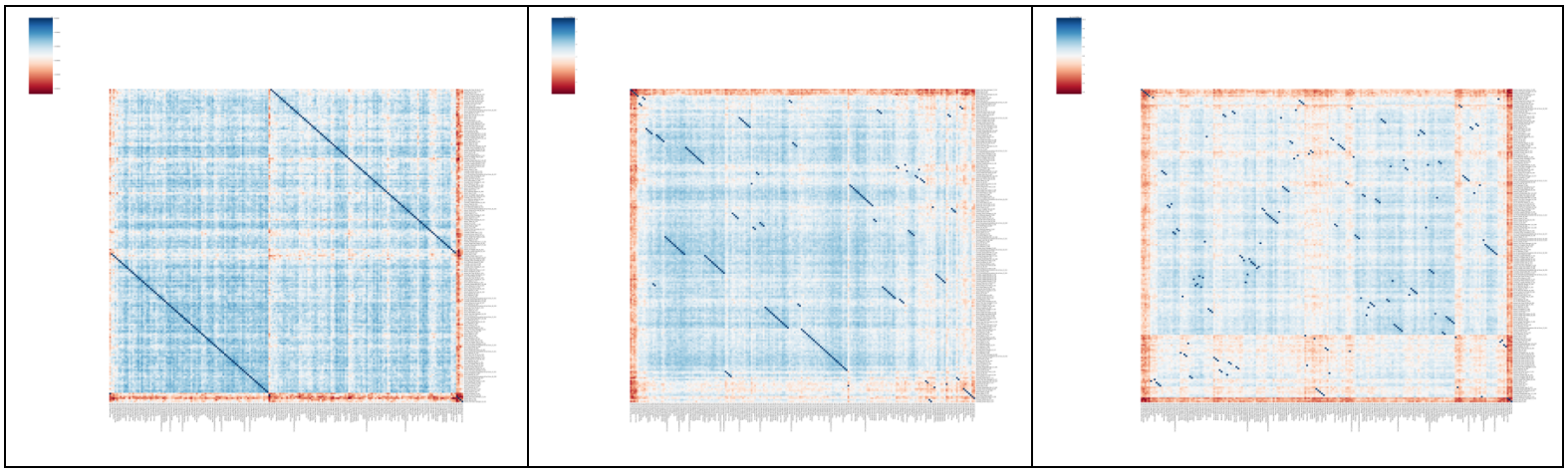
Data Wrangling Method2: baseline + lemmatization + stopwards			
Data Wrangling Method3: baseline + lemmatization + stopwards + alpha only			

The figure 9 depicts doc2vec algorithm's tsne plot analysis for three distinct data wrangling techniques and embedding sizes of 100, 200, and 300. The tsne-plots is a 2-dimensional scatter plot in which we are attempting to examine and assess if these 100 random words shown contain clusters of similar terms from different movie reviews or genres using doc2vec techniques. However, the t-SNE charts for baseline plot doesn't appear to have clearly defined clustering since it contains a lots of common stop words. The method 2 has more defined shapes represent clustering for all embedding sizes compared to method 1 but not consistent across the broad. For example: some terms from horror and comedy are common among the reviews for movie like family might be correlating 'Dirty Grandpa', 'Grown Ups', and 'IT Chapter Two' which make sense but while analyzing other clusters wasn't showing consistent

results. Similarly, method 3 had much better results with proper clustering compared to method 2 as the embedding size was increased to 300 but having a larger class corpus size would really help in visualizing the clustering of t-SNE plot using doc2vec algorithm.

Figure 10: Doc2Vec Chart HeatMap (All 9 Experiments)

BL_Doc2Vec_100	BL_Doc2Vec_200	BL_Doc2Vec_300
		
M2_Doc2Vec_100	**M2_Doc2Vec_200**	**M2_Doc2Vec_300**
		
M3_Doc2Vec_100	**M3_Doc2Vec_200**	**M3_Doc2Vec_300**



The figure 10 shows doc2vec algorithm heat map clustering analysis is based on cosine similarity for three different data wrangling methods on 100 words for all 9 experiments, and different embedding sizes of 100, 200, and 300. Similar to Word2Vec heatmap analysis the observation is difficult to visibly analyze a well-defined cluster as most of the charts in Method 2 and Method 3 with embedding size 100, 200, and 300 shows colored light to heavy blue which coincides with cosine similarity of 1. Based on our analysis well-defined clustering map the second data wrangling method and word embedding size of 100 which is M2_Doc2Vec_100. Hence, the word clustering doesn't appear to be consistent for Method 2 and 3 charts as we increasing the embedding size from 100 to 300. According to me, word2vec cluster map analysis was more useful compared to doc2vec algorithm for this class corpus since as we increase the embedding size the scatter plot are different small diagonal lines cross the dimensional space with no consistency based on the observation.

Conclusion

The qualitative experimentation results clarified a lot from the tf-idf score perspective for all the reviews documents are across one movie first, so we can gather better insights from 10 movie review documents. Before deep diving into the quantitative approach experimentation with entire class corpus data using different data wrangling methods and analyzing using

word2vec and doc2vec algorithm with visualization based on heatmaps, cluster maps, and t-SNE plots. The most common words in the movie ‘Us’ that appeared most frequently in the reviews after conducting tf-idf vectorization experiment were scary, film, family, and us. In addition, some words were also found in "all horror" reviews, but certain terms were in "all non-horror" reviews as well. Based on the results following the quantitative approach, I would recommend using tf-idf vectorization with data wrangling method 2 because once lemmatization and removal of stop words from the class corpus data was done it provided better classification and visible clustering for tf-idf cosine-similarity heatmaps as well as for both the t-SNE and clustering heatmap for different movie review across each genre. However, after applying the data wrangling method 2 of doing lemmatization and removal of stop words the word2vec algorithm clustering heatmap with embedding size 200 shows better visualization middle of chart showing similar words such as emotion, strange, social, mother, shadow, and vacation shows is closely related to different genre and movie in the same context. In conclusion, both word2vec and doc2vec t-SNE plots were not plotting well-defined visible 20 clusters which we were looking for from this analysis as it requires further experimentation on both increasing of random words and embedding size that is being used plotting the t-SNE plot as well as the size of the class corpus data.

References

Jurafsky, D. and Martin, J., 2019, Speech and Language Processing (3rd ed.draft)

Lane, H., C. Howard, and H. M. Hapke 2019. Natural Language Processing in Action: Understanding, Analyzing, and Generating Text with Python.

Python Wife. 2022. TF-IDF vectors in Natural Language Processing. Retrieved from https://pythonwife.com/tf-idf-vectors-in-natural-language-processing/