# NLP CLUSTERING, CLASSIFICATION, AND TOPIC MODELING OF MOVIE REVIEWS

Betzalel Moskowitz
MSDS453-DL: Natural Language Processing
November 17, 2023

## Introduction

While rule-based systems were the focus of early natural language processing (NLP) systems, these systems were often brittle, tedious to maintain, and struggled to model semantic meaning of text. The advent of vectorized representations enabled the application of both supervised and unsupervised machine learning algorithms to difficult business tasks such as topic modeling, clustering, information retrieval, sentiment analysis, and multi-class classification. This paper explores using various machine learning methods in NLP on a dataset consisting of movie reviews across different genres. The dataset, comprised of twenty movies spanning four genres, serves as the foundation for experiments in clustering, binary and multi-class classification, and topic modeling. The paper evaluates the effectiveness of different vectorization techniques, including TF-IDF and Doc2Vec embeddings, in capturing semantic information and aiding machine learning algorithms in tasks such as sentiment analysis and genre classification. Additionally, topic modeling is explored using Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) to assess the feasibility of grouping together documents by topic.

## Methods

*Dataset*

The corpus for these experiments is comprised of twenty different movie reviews of movies across four different genres ("Action", "Comedy", "Horror", "Sci-Fi"). These movie reviews were sourced from *Rotten Tomatoes* and *IMDB.* For each of the twenty movies, ten reviews were selected – five contained "positive" sentiment and the other five contained "negative" sentiment. This resulted in a corpus of 200 movie reviews which are referred to in this report as the *documents*.

*Data Preparation*

Each document was truncated at 500 tokens. The text from each document was normalized by removing punctuation, converting to lowercase, removing any special HTML tags, and removing special characters and digits. The documents were then tokenized, and stop words were removed in an attempt to remove noisy words that provided little semantic value to the corpus. The stop words consisted of some of the most common English words as well as several custom stop words ('movies', 'movie', 'film', 'films', 'scene') that were prevalent across all documents and seemed to add noise that made NLP tasks more difficult. Finally, all tokens were lemmatized using NLTK's WordNetLemmatizer (NLTK).

*Vectorization*

With the data preparation complete, the documents were vectorized into numerical representations for the machine learning algorithms to operate over. With the goal in mind to analyze the performance of different vectorization techniques and representations, the documents were transformed into TF-IDF vectors using sklearn's TfidfVectorizer (Scikit-Learn Developers) and Doc2Vec embeddings using gensim's Doc2Vec (Řehůřek, 2022). To examine the role of

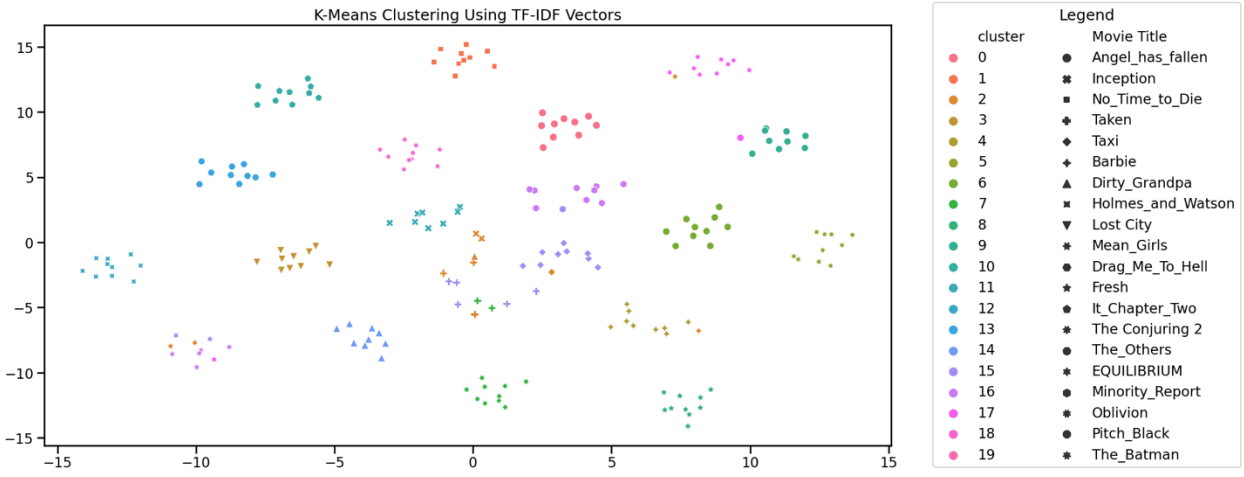
different embedding sizes in the vector representations, Doc2Vec experiments were running with embedding sizes of 100, 200, and 300 for clustering experiments. Doc2Vec embedding sizes 100, 500, and 1000 were experimented with for classification experiments.

*Clustering*

Clustering experiments were conducted to assess the efficacy of grouping together similar documents, a useful capability for building information retrieval and recommendation systems. K-Means clustering models were trained for each of the four vectorization methods (TF-IDF vector, and Doc2Vec sizes 100, 200, 300).

For reproducibility purposes, all models were trained with scikit-learn using the same random state (20130810). The number of clusters were determined for each vectorization technique by training K-Means clustering models with *k* values of four through twenty and selecting the *k* value of the model yielding the highest silhouette score. In order to generate a visualization to assess the quality of the clusters, T-SNE models were fit with two components in order to reduce the dimensionality of the document vectors into a two-dimensional space that could be visualized in a two-dimensional plot. The T-SNE-transformed document vectors were then used to generate a scatterplot where each document was a point, colored according to the cluster that the K-Means model assigned it to. To assess the quality of these clusters, the points were styled in unique shapes that corresponded to a certain supervised label associated with the document such as genre or movie title. This allowed for the ability to qualitatively assess the clustering performance through determining whether certain clusters appeared to be comprised of documents with the same labels (stronger clusters) or a mix of labels (weaker clusters). It also allowed for the ability to interpret the characteristics of specific clusters (ex. clusters with documents of one specific movie/genre). *Figure 1* provides an example of a visualization generated to assess how well the twenty-cluster TF-IDF model groups together documents by movie titles. All clustering plots are provided in *Appendices 1-3*.

***Figure 1. Visualization of K-Means Model for TF-IDF-Vectorized Documents by Movie Title***



*Binary Classification*

Many supervised learning NLP applications are binary – they attempt to answer the question "yes" or "no", "positive" or "negative" and are used in predictive modeling applications such as spam detection systems, fraud detection systems, and in our case – binary sentiment analysis. To determine the efficacy of using classical machine learning techniques on TF-IDF vectors and different-sized Doc2Vec embeddings (100, 500, 1000) for sentiment classification, models were built to predict the sentiment of each document. To do this, a test set consisting of 33% of the dataset was held out to assess performance.

For each vectorization method (TF-IDF, Doc2Vec embedding sizes 100, 500, 1000), a support vector machine (SVM), Decision Tree, and a Random Forest model were trained to predict whether a vectorized document had "positive" or "negative" sentiment. All models were implemented using Python's open-source machine learning library – scikit-learn. Limited hyperparameter-tuning was conducted to select the hyperparameters for the best model. Because the dataset was well-balanced between sentiment classes (roughly a 50-50 split in both training and test sets), highest test accuracy was selected as the criteria to identify the best performing models. For SVM models, this involved tuning the $C$ value ($C \in \{0.01, 1, 1, 10, 100, 1000, 10000\}$). For Decision Tree models, max depth was tuned ($max\_depth \in \{1, 2, \ldots, 19\}$). Random Forest models were tuned by selecting different numbers of trees in increments of ten ($n\_estimators \in \{10, 20, 30, \ldots, 200\}$), also selecting the best model according to test accuracy. All models used the same random state for reproducibility purposes.

Confusion matrices were produced, and test accuracy, test precision, test recall, and test F1 were computed and recorded to assess the performance of the different models. The confusion matrices for the binary classification problem can be found in *Appendix A4* and the performance metrics can be found in *Appendix A5*.

*Multi-Class Classification*

Other supervised learning systems are trained for multi-class classification tasks. Instead of predicting the target variable from a set of only two targets, multi-class classification tasks involve predicting a target variable from a set of *more than* two targets. NLP applications that involve this sort of task include language identification, sentiment analysis with multiple emotions, and topic classification in text content using pre-identified targets. To assess the efficacy of using classical machine learning techniques on TF-IDF vectors and different-sized Doc2Vec embeddings (100, 500, 1000) for multi-class classification tasks, models were built to predict the genre from a given document. This is a multi-class classification problem as it involves predicting the correct genre from a set of four genres in the dataset ("Action", "Comedy", "Horror", "Sci-Fi").

The same dataset splitting, models, and training methodologies as the binary classification were used for this task, only this time training the models to predict the genre. Confusion matrices were produced, and test accuracy, test precision, test recall, and test F1 were computed and recorded to assess the performance of the different models on the genre classification task. The

confusion matrices for the multi-class classification problem can be found in *Appendix A6* and the performance metrics can be found in *Appendix A7*.

*Topic Modeling*

Topic modeling is another useful technique that can be applied to identify topics within a collection of documents. It can be used for document clustering, content recommendation, information retrieval, social media analysis, customer reviews analysis, news article categorization, and much more.

Two topic modeling algorithms were used in experiments intended to group together movie review documents with similar topics – Latent Semantic Analysis/Latent Semantic Indexing (LSA/LSI) and Latent Dirichlet Allocation (LDA).

LSA models are designed to capture the latent structure of a document term-matrix (derived from the corpus) through singular value decomposition. It involves selecting the number of concepts to capture and the number of words per topic. The LSA models were trained using gensim's LsiModel (Řehůřek, 2022). For simplification purposes, only ten words per topic were used when training these models.

LDA models were also used in an attempt to probabilistically model the topics associated with each document. This method made use of only the TF-IDF document vectors and gensim's LdaModel (Řehůřek, 2022) was used for implementation. For simplification purposes, 20 words per topic were used when training LDA models.

To assess the best number of concepts/topics for both the LSA and LDA models, the models were both trained using values 2-19 for number of topics/clusters. For each model trained, a coherence score was computed using the "c_v" method to determine the interpretability of each concept/topic. The best model was determined as the model with the highest coherence score.

To understand the topics/concepts, each model was printed out as a list of tuples, with each tuple representing a topic/concept. The tuples' first elements yielded the topic number and strings as its second element, similar to the one in *Figure 2*. The string displays the words related to the topic and a weight assigned to each word to signify the word's weight within the topic.

***Figure 2. Topics from a 2 Topic, 10 Word LSA Model***

*[(0, '0.003*"action" + 0.003*"story" + 0.003*"horror" + 0.003*"house" + 0.003*"first" + 0.003*"characters" + 0.002*"character" + 0.002*"school" + 0.002*"years" + 0.002*"still"), (1, '0.004*"would" + 0.003*"barbie" + 0.003*"holmes" + 0.003*"world" + 0.003*"first" + 0.002*"story" + 0.002*"watson" + 0.002*"comedy" + 0.002*"action" + 0.002*"years")]*

To visualize how well the LSA/LDA models group together similar documents, a custom function was developed to create a heatmap of the similarity of documents. The custom function calculates the similarity of topics between documents using the LSA/LDA models and visualizes

the similarity matrix as a heatmap. The x and y-axis labels of the heatmap correspond to the titles of the documents, and the color intensity represents the degree of similarity between the topcis of the documents. These heatmaps for the different LSA models can be seen in *Appendices A8-A11* and the different LDA models can be found in *Appendices A12-A14*.

## Results

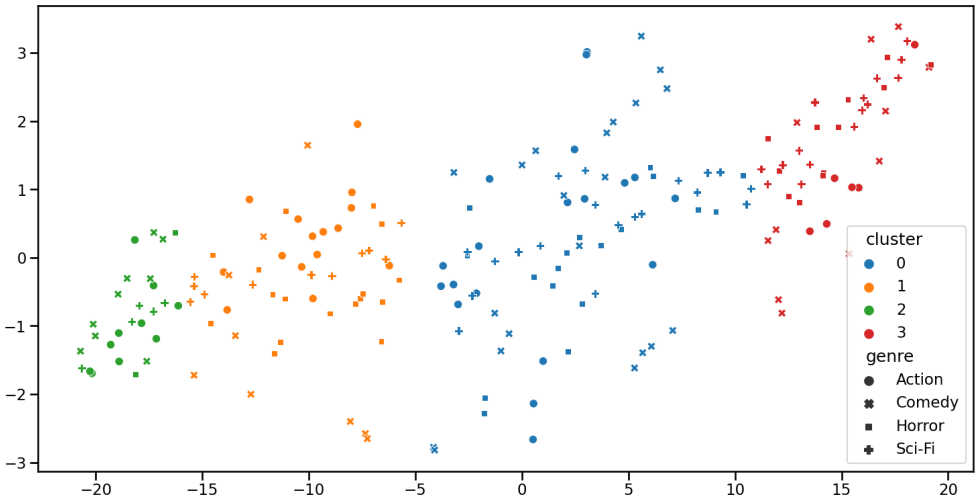
*Clustering*

After conducting the clustering experiments, it was evident that the TF-IDF clustering model with twenty clusters was the most successful at clustering the data. This can be seen qualitatively in *Appendix A1*, where the labels making up each cluster were almost completely homogeneous across all twenty movie titles. Impressively, the twenty-cluster model proved successful at producing clusters that almost always contained ten points with all of the same movie titles. This model had a silhouette score of 0.0823, the highest silhouette score among all K-means clustering models for TF-IDF vectors.

The other TF-IDF model with four clusters performed poorly, with a silhouette score of 0.0197. Nonetheless, this number of clusters was chosen to examine whether four clusters could segment the documents according to genre. The poor performance can be visualized in *Appendix A2*, where numerous genres appear in each cluster, suggesting that the clusters did not succeed at segmenting the documents according to genre, but perhaps by a different criterion unknown to humans.

The Doc2Vec K-means models were more difficult to assess –the best Doc2Vec model (size 300) yielded a silhouette score of 0.4056 and the Doc2Vec clusters had higher silhouette scores for four-cluster K-means clusters than their TF-IDF counterparts. However, these clusters (visualized in *Figure 3*) did not seem to perform particularly well at clustering the documents in a way that segmented the documents according to genre. This was evident as none of the genres were concentrated *primarily* in specific clusters.

***Figure 3. Evaluating K-Means Model on Doc2Vec Size 300 for Clustering by Genre***



*Binary Classification (Sentiment Analysis)*

Sentiment analysis for this dataset proved rather challenging. Looking at the table of performance metrics in *Appendix A5*, one can see that the performance was quite disappointing. The best test accuracy achieved by any model was 0.61 for the Doc2Vec (size 100) Random Forest model. Overall, the SVM and Decision Tree models seemed to largely predict a single class (see *Appendix A4)*, resulting in imbalanced performance metrics among the classes. For example, the SVM model for the TF-IDF vectors achieved a recall of 0.82 for negative reviews but scored a disappointing 0.27 recall on positive classes. The Random Forest models overall seemed to achieve better performance on this task by the metric of test accuracy, likely due to the of ensemble methods strength of avoiding overfitting and leveraging the "wisdom of the crowds".

*Multi-Class Classification (Genre Recognition)*

Performance on the genre recognition task was significantly higher than on the binary sentiment prediction task. The SVM models in particular stole the show – the TF-IDF SVM model scored a perfect precision, recall, F1, and accuracy on the test set. The SVM models for the Doc2Vec vectors of size 500 and 1000 both scored a near-perfect 0.98 test accuracy. Decision trees performed very poorly across all vectorization techniques but performed the best (0.61 test accuracy) on the TF-IDF vectors. The random forest models outperformed the decision tree models overall but performed significantly worse on Doc2Vec models (0.63 test accuracy on average across all three) than the TF-IDF vectorized documents (0.94 test accuracy). Overall, the TF-IDF vectors appeared to be the superior vectorization method for this task while the best machine learning algorithm run in these experiments was the SVM. A larger embedding size did not seem to consistently produce better results for the Doc2Vec models.

*Topic Modeling*

The topic modeling results for LSA showed that a suitable number of concepts are necessary to group together similar documents by topics/concepts. As can be seen in the first cell of *Figure 4*, having too few concepts results in concepts that don't help in grouping together multiple documents with similar concepts as all documents appear too similar to each other. The four-concept LSA model, while having a lower coherence score, was slightly better at segmenting documents – the visualization in the second cell in *Figure 4* demonstrates that documents conceptually close to each other are brighter and documents that were conceptually different were darker. The increased amount of contrast suggests that the model is learning the patterns to group by concepts rather than claiming that every document is conceptually close to one another. That said, the twenty-concepts LSA model was the best topic model produced during experimentation – it had one of the highest coherence scores at approximately 0.5010, and also seemed to clearly identify concepts that are unique to only documents with the same movie titles. This can be seen in the visualization as the only bright spots appeared when the similarity of a document is compared to another document with the same movie title. *Figure 5* shows an example of a concept and the weights of the words that belong to it. The concept in *Figure 5* seems to represent the movie "Equilibrium".

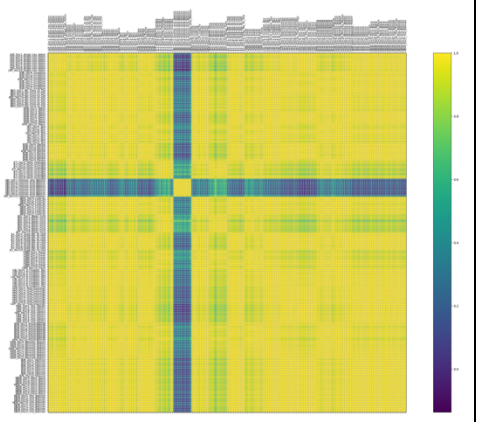
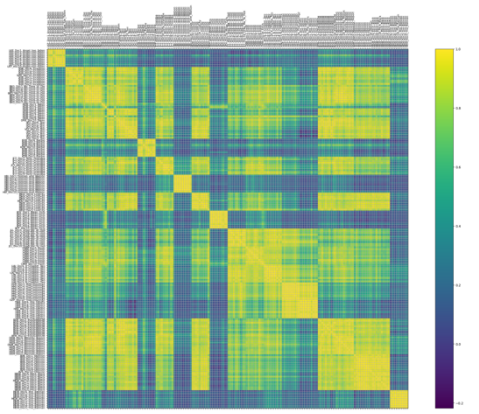
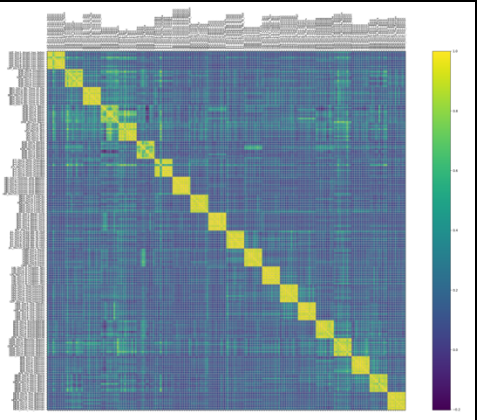
| *Figure 4. LSA Similarity Heatmaps* | | |
|---|---|---|
| **2 Concepts, 10 Words** | **4 Concepts, 10 Words** | **20 Concepts, 10 Words** |
| Coherence = 0.4837 | Coherence = 0.3764 | Coherence = 0.5010 |
|  |  |  |

***Figure 5. Words and of LSA Model on Concept Related to Movie "Equilibrium"***

*'0.319*"preston" + 0.257*"emotions" + 0.189*"equilibrium" + -0.167*"inception" + -0.160*"nolan" + -0.130*"oblivion" + 0.129*"world" + 0.120*"called" + 0.119*"christian" + -0.115*"story"'*

The LDA models performed poorly compared to the LSA models. *Figure 6* shows that the coherence scores of the LDA models were significantly lower, and the brighter portions in the heatmaps are a lot more sporadic and difficult to interpret. In comparing the best LSA model (20 concepts) and the best LDA model (18 concepts), we can see that both have the diagonal line indicating that the documents are similar to each other in terms of topic/concept. However, the diagonal line is thinner in the LDA models, indicating that the LDA model did not perform as well in identifying that other documents with the same movie title are close in terms of topic.

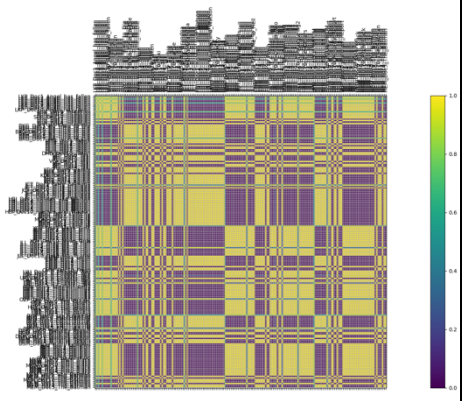
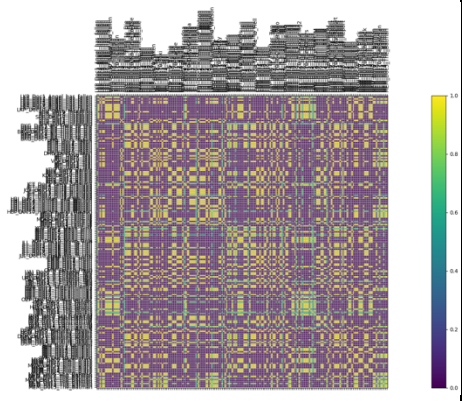
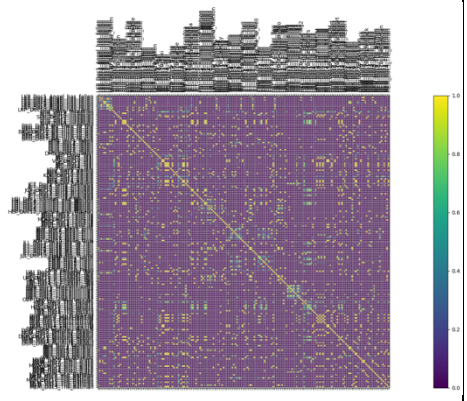
| *Figure 6. LDA Similarity Heatmaps* | | |
|---|---|---|
| **2 Topics, 20 Words** | **4 Topics, 20 Words** | **18 Topics, 20 Words** |
| Coherence = 0.2113 | Coherence = 0.2219 | Coherence = 0.3195 |
|  |  |  |

*Figure 7* shows an example of a topic and the weights of the words that belong to it for the best LDA model. The topic in *Figure 7* seems to describe the movie "*Equilibrium*". However, it should be noted that while some "*Equilibrium"*-related terms are found in Figure 7, the coherence weights for these words are significantly lower than in the LSA model, indicating that the words in these topics were less closely related than in the LSA models.

***Figure 7. Words and of LDA Model on Concept Related to Movie "Equilibrium"***

*'0.004*"really" + 0.004*"something" + 0.004*"future" + 0.004*"character" + 0.003*"preston" + 0.003*"takes" + 0.003*"characters" + 0.003*"world" + 0.003*"equilibrium" + 0.003*"either"'*

## Analysis

While the twenty-cluster TF-IDF model appeared to perform well at clustering documents according to movie title, the rest of the clustering models largely struggled to cluster documents according to supervised labels, such as genre. While unsupervised methods like clustering can be used to group similar documents together, they do so without knowledge of any labels and are focused on discovering natural patterns or groupings in the data, which may not necessarily correspond to supervised classes. The data also typically contain inherent complexities or substructures that are not captured by a simple mapping to the predefined classes. Thus, using only the number of supervised target classes as the number of classes may oversimplify the grouping and miss the finer patterns in the data typically uncovered by unsupervised learning approaches. While performance of unsupervised clustering algorithms is more difficult to assess quantitatively than supervised algorithms, it is better to choose the model with the best unsupervised learning metrics (like silhouette scores) and then apply qualitative measures to assess whether the clusters seem to be aligned with any of the predefined classes. There may not be exactly the same number of clusters as there are target classes but allowing the unsupervised algorithm the freedom to use any *k* number of clusters allows the unsupervised algorithm to do its best to uncover patterns. This can be difficult to remember when focusing on predicting predefined classes.

The disappointing performance on the binary sentiment classification task appears unrelated to the selection of algorithm. Instead, this seems to be linked to a data quality issue. From previous experience selecting movie review documents to add to the corpus, a noticeable pattern emerged in movie reviews – authors generally summarize the movies in the beginning of the review before providing their critiques. Given that each document was truncated at 500 words during the building of the corpus, many documents may have had their critical elements removed from the text. If a human were to then attempt this task without having access to the critical portions of the document, they would also likely struggle to identify the sentiment in the review. Thus, it is likely that the important features typically used to classify sentiment were not present in the data and resulted in poor performance. This underscores the importance of exploring the data before making data wrangling decisions that may potentially hinder performance.

The strong performance on the genre classification using the SVM algorithms indicated that classic machine learning approaches may indeed perform well on multi-class classification tasks. The results also uncovered that the TF-IDF vectors were more performant than the Doc2Vec vectors. This trend appeared across multiple NLP tasks in these experiments. While Doc2Vec embeddings are more sophisticated and context-specific, the small corpus size (200 documents) may have been insufficient to produce a strong embedding space – after all, Doc2Vec models are neural network-based and rely on large amounts of data to perform well.

It was unclear whether a larger Doc2Vec embedding size benefitted performance – performance on larger embedding sizes was not consistently higher or lower than smaller embedding sizes. A larger dataset may be needed to improve performance rather than only changing embedding sizes.

For topic modeling, the LSA models seemed to outperform the LDA models both in the quality and interpretability of the topics/concepts. This was likely due to several factors. First, LSA models generally provide a more straightforward and transparent representation of topics by capturing the latent structure of the document-term matrix through singular value decomposition (SVD), making it easier to understand and interpret the identified topics. LSA models also tend to be more performant on smaller datasets such as the one used in these experiments, especially when the Dirichlet distribution assumption is not necessarily followed. The stronger LSA performance may also have occurred given that TF-IDF vectors seemed to perform better on these experiments and LSA uses a TF-IDF weighting scheme. LSA also seems to work better with smaller datasets and a sparser document-term matrix that were characteristic of this problem. It would be interesting to run these experiments again on a much larger dataset to see if the LDA models can outperform the LSA models with access to more data and dense representations.

It’s also worth noting that many of these models achieved poorer performance until the most common and semantically noisy words were removed as custom stop words. This allowed the models to focus on tokens that provided semantic value for these tasks and less on those that largely added noise.

The experiments in this study serve as a testament to the data-centric nature of NLP problems, emphasizing the need for careful consideration of dataset characteristics. Future work could explore sentiment classification on full-length movie reviews, potentially alleviating the challenges observed in this study. Additionally, utilizing larger datasets may enhance the performance of Doc2Vec embeddings and enable the exploration of deep learning approaches in tackling NLP tasks related to movie reviews.

## Conclusion

The studies in this paper uncovered nuances in the performance of various NLP methods, shedding light on both their strengths and limitations. The clustering experiments revealed that the TF-IDF models excelled in grouping similar documents, especially when assessing sentiment through movie titles. However, the binary sentiment classification task posed challenges, hinting

at potential data quality issues such as the truncation of reviews. The multi-class classification task, focusing on genre recognition, showcased the supremacy of Support Vector Machine (SVM) models, particularly when applied to TF-IDF vectors. Surprisingly, Doc2Vec embeddings did not outperform TF-IDF vectors, suggesting that the small corpus size may have limited the effectiveness of the neural network-based Doc2Vec models. Topic modeling experiments underscored the dominance of LSA over LDA in terms of quality and interpretability of identified topics. LSA's transparent representation and compatibility with smaller datasets played pivotal roles in its success. Notably, the removal of common and semantically noisy words as custom stop words significantly improved the models' performance, emphasizing the importance of meticulous data preprocessing in NLP tasks.

## References

*nltk.stem.wordnet module*. NLTK. (n.d.). https://www.nltk.org/api/nltk.stem.wordnet.html

Řehůřek, R. (2022, December 21). *Gensim: Topic modelling for humans*. models.ldamodel – Latenet Dirichlet Allocation - gensim. https://radimrehurek.com/gensim/models/ldamodel.html

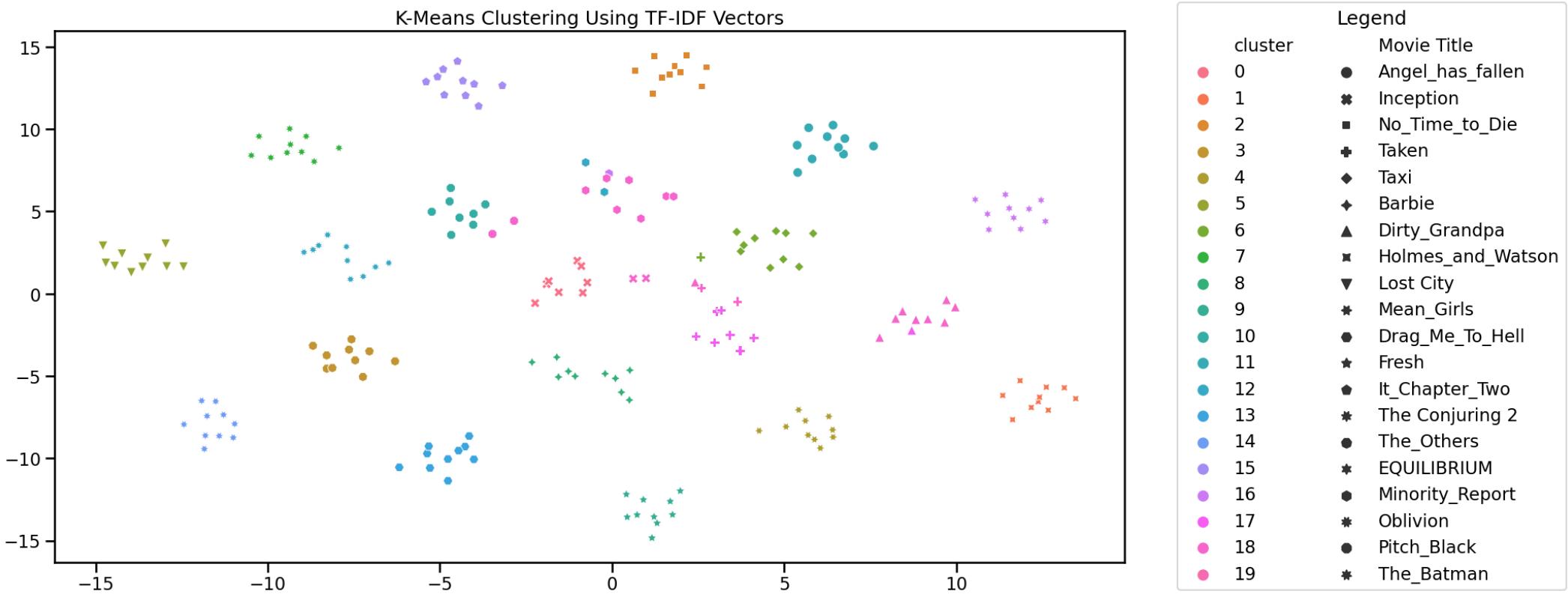
Řehůřek, R. (2022, December 21). *Gensim: Topic modelling for humans*. models.lsimodel – Latenet Semantic Indexing - gensim. https://radimrehurek.com/gensim/models/lsimodel.html

Řehůřek, R. (2022, December 21). *Gensim: Topic modelling for humans*. Doc2Vec Model - gensim. https://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html

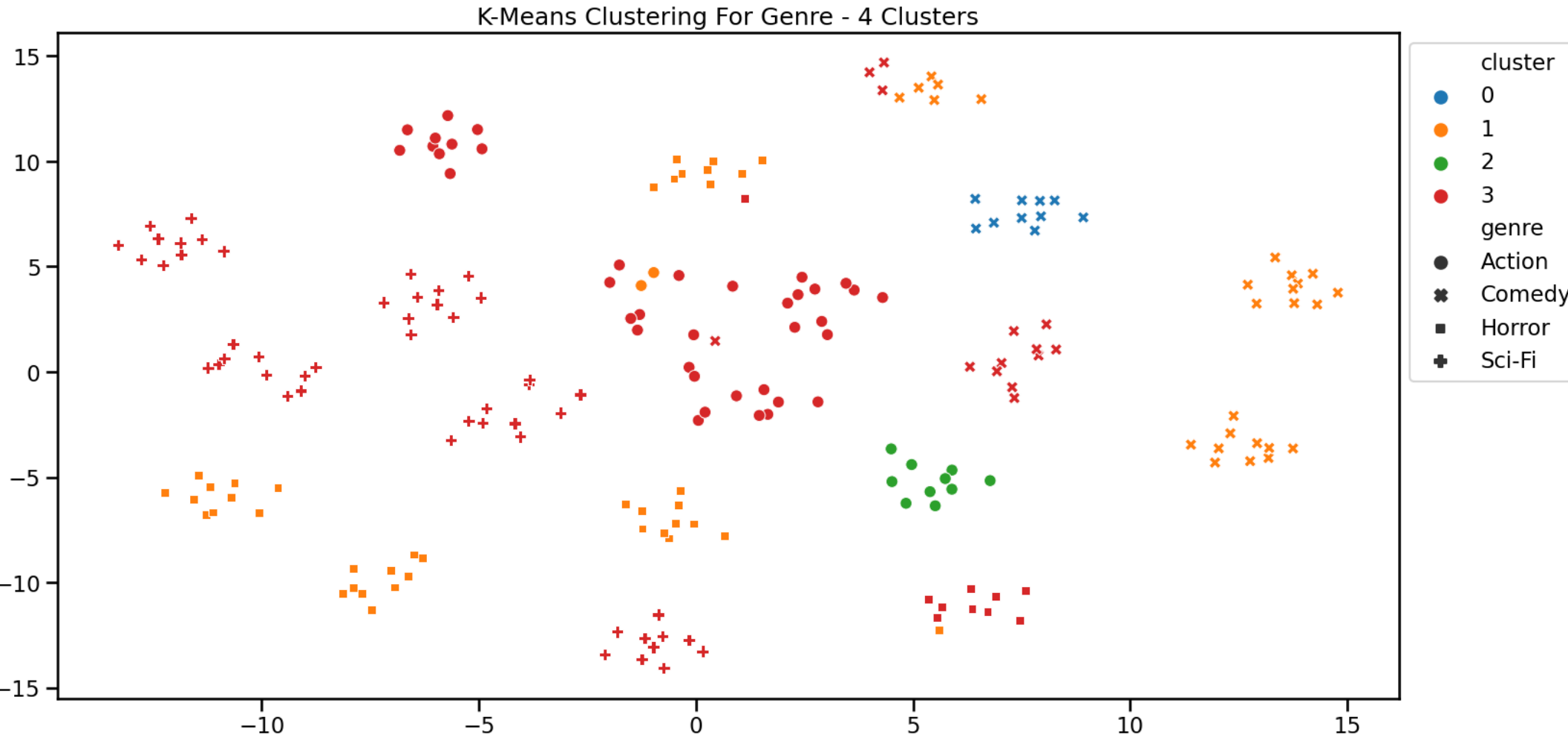
Scikit-Learn Developers. (n.d.). *Sklearn.feature_extraction.text.TfidfVectorizer*. scikit. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

## Appendix

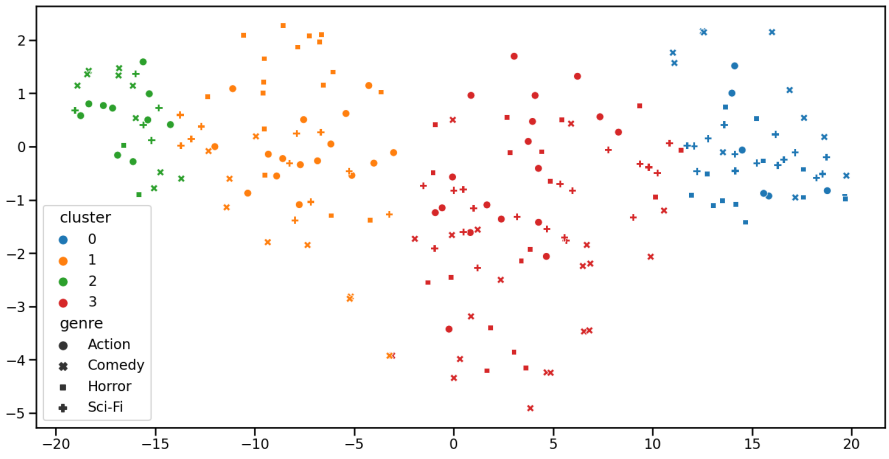
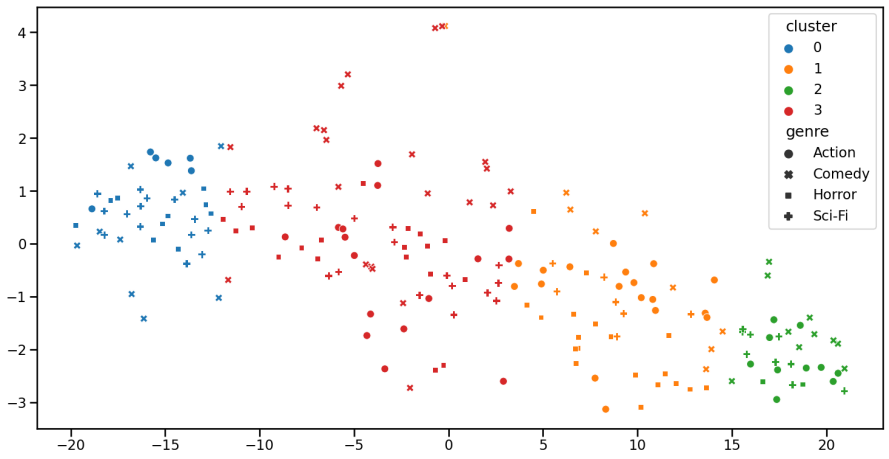
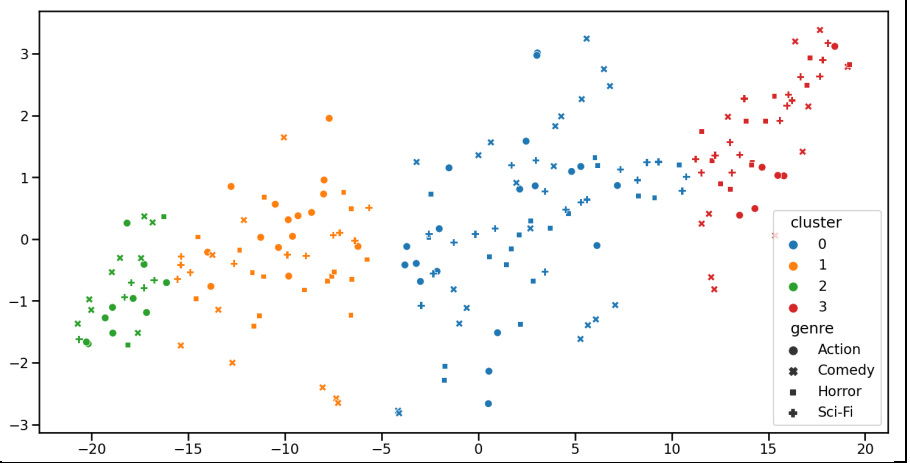
### A1. PCA-Reduced Clustering by Movie Using TF-IDF Vectors – 20 Clusters



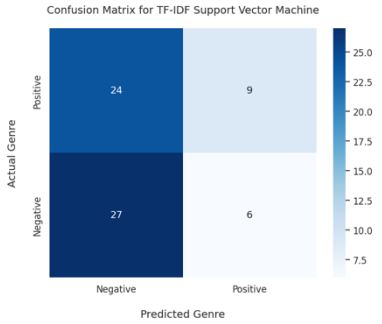
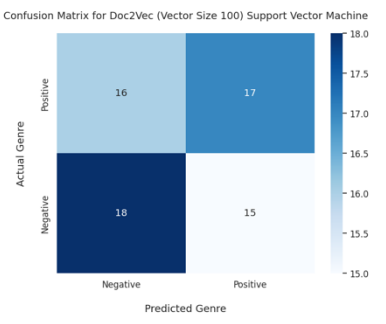
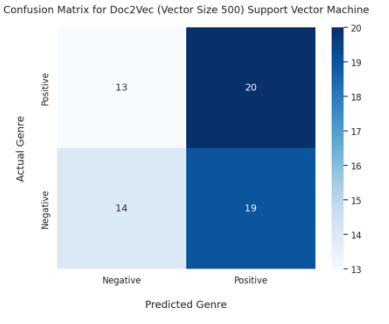
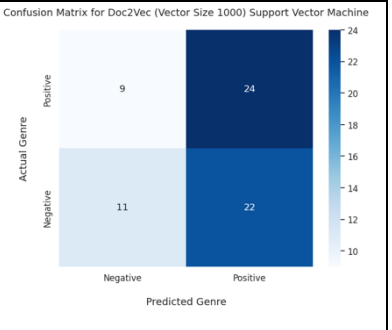
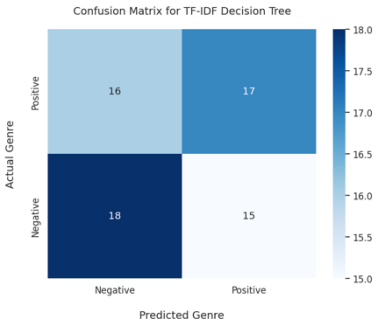
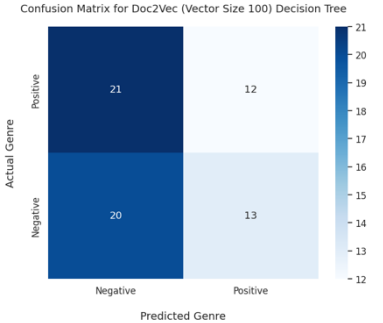
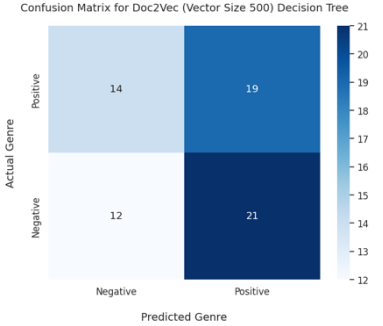
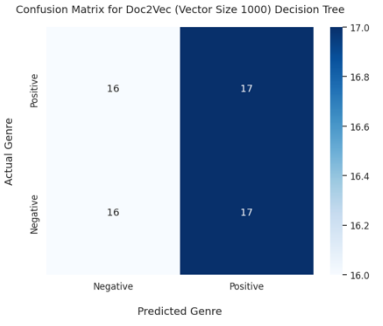
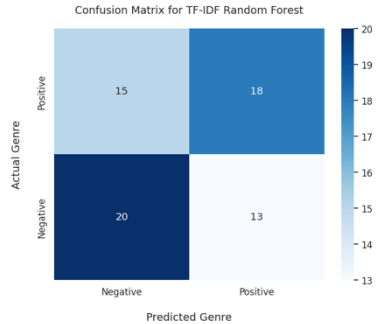
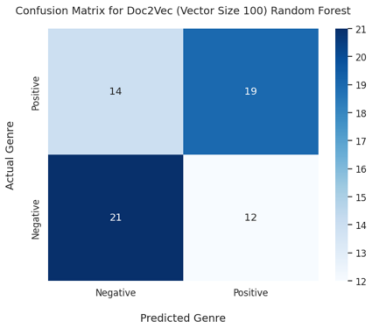
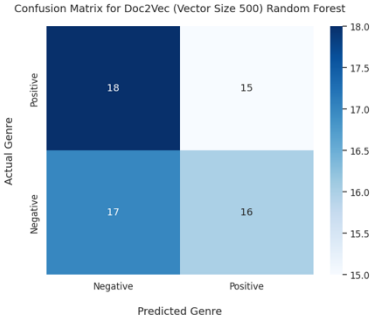
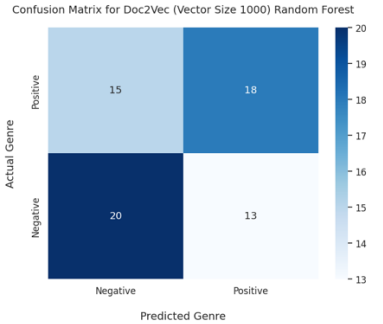
**A2. PCA-Reduced Clustering By Genre Using TF-IDF Vectors – 4 Clusters**



## A3. PCA-Reduced Clustering Using Doc2Vec Embeddings

| Embedding Size | Cluster Visualizations |
|---|---|
| 100 |  |
| 200 |  |
| 300 |  |

# A4. Sentiment Analysis Confusion Matrices

| Model | TF-IDF | Doc2Vec Size 100 | Doc2Vec Size 500 | Doc2Vec Size 1000 |
|---|---|---|---|---|
| **Support Vector Machine** |  |  |  |  |
| **Decision Tree** |  |  |  |  |
| **Random Forest** |  |  |  |  |

## A5. Review Sentiment Classification Performance Metrics

### Support Vector Machine

**TF-IDF**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.53 | 0.82 | 0.64 | 33 |
| Positive | 0.60 | 0.27 | 0.37 | 33 |
| Accuracy | | | 0.55 | 66 |
| Macro Avg | 0.56 | 0.55 | 0.51 | 66 |
| Weighted Avg | 0.56 | 0.55 | 0.51 | 66 |

**Doc2Vec Size 100**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.53 | 0.55 | 0.54 | 33 |
| Positive | 0.53 | 0.52 | 0.52 | 33 |
| Accuracy | | | 0.53 | 66 |
| Macro Avg | 0.53 | 0.53 | 0.53 | 66 |
| Weighted Avg | 0.53 | 0.53 | 0.53 | 66 |

**Doc2Vec Size 500**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.52 | 0.42 | 0.47 | 33 |
| Positive | 0.51 | 0.61 | 0.56 | 33 |
| Accuracy | | | 0.52 | 66 |
| Macro Avg | 0.52 | 0.52 | 0..51 | 66 |
| Weighted Avg | 0.52 | 0.52 | 0.51 | 66 |

**Doc2Vec Size 1000**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.50 | 0.50 | 0.49 | 33 |
| Positive | 0.50 | 0.52 | 0.51 | 33 |
| Accuracy | | | 0.50 | 66 |
| Macro Avg | 0.50 | 0.50 | 0.50 | 66 |
| Weighted Avg | 0.50 | 0.50 | 0.50 | 66 |

### Decision Tree

**TF-IDF**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.53 | 0.55 | 0.54 | 33 |
| Positive | 0.53 | 0.52 | 0.52 | 33 |
| Accuracy | | | 0.53 | 66 |
| Macro Avg | 0.53 | 0.53 | 0.53 | 66 |
| Weighted Avg | 0.53 | 0.53 | 0.53 | 66 |

**Doc2Vec Size 100**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.49 | 0.61 | 0.54 | 33 |
| Positive | 0.48 | 0.36 | 0.41 | 33 |
| Accuracy | | | 0.48 | 66 |
| Macro Avg | 0.48 | 0.48 | 0.48 | 66 |
| Weighted Avg | 0.48 | 0.48 | 0.48 | 66 |

**Doc2Vec Size 500**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.46 | 0.36 | 0.41 | 33 |
| Positive | 0.47 | 0.58 | 0.52 | 33 |
| Accuracy | | | 0.47 | 66 |
| Macro Avg | 0.47 | 0.47 | 0.46 | 66 |
| Weighted Avg | 0.47 | 0.47 | 0.46 | 66 |

**Doc2Vec Size 1000**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.57 | 0.61 | 0.59 | 33 |
| Positive | 0.58 | 0.55 | 0.56 | 33 |
| Accuracy | | | 0.58 | 66 |
| Macro Avg | 0.58 | 0.58 | 0.58 | 66 |
| Weighted Avg | 0.58 | 0.58 | 0.58 | 66 |

### Random Forest

**TF-IDF**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.57 | 0..61 | 0.59 | 33 |
| Positive | 0.58 | 0.58 | 0.58 | 33 |
| Accuracy | | | 0.58 | 66 |
| Macro Avg | 0.58 | 0.58 | 0.58 | 66 |
| Weighted Avg | 0.58 | 0.58 | 0.58 | 66 |

**Doc2Vec Size 100**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.60 | 0.64 | 0.62 | 33 |
| Positive | 0.61 | 0.58 | 0.59 | 33 |
| Accuracy | | | 0.61 | 66 |
| Macro Avg | 0.61 | 0.61 | 0.61 | 66 |
| Weighted Avg | 0.61 | 0.61 | 0.61 | 66 |

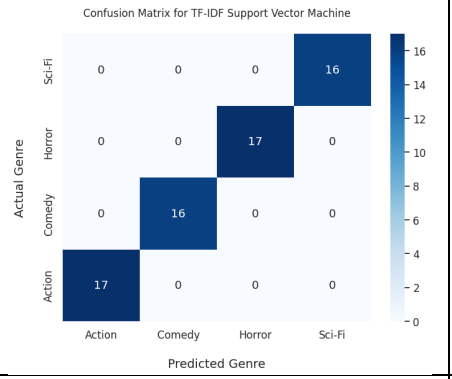
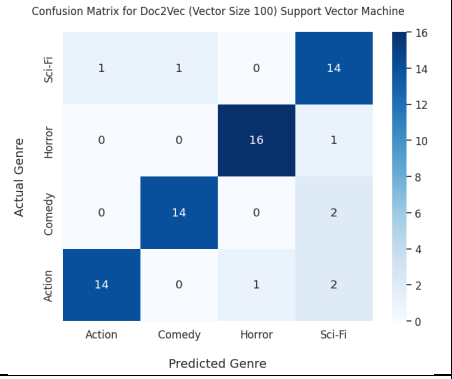
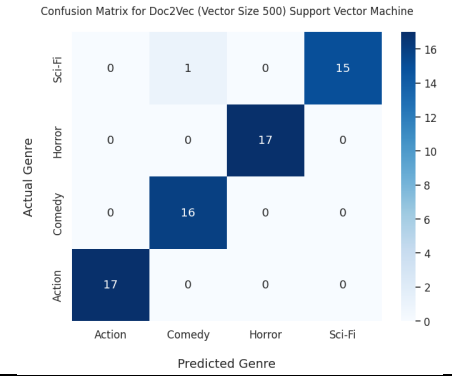
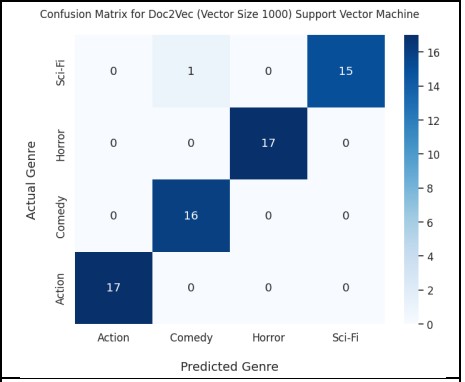
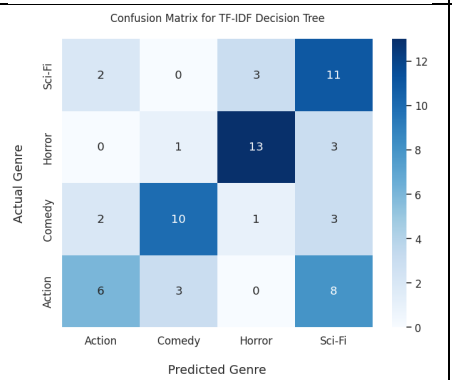
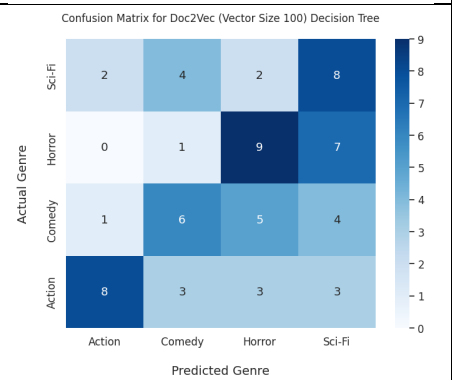
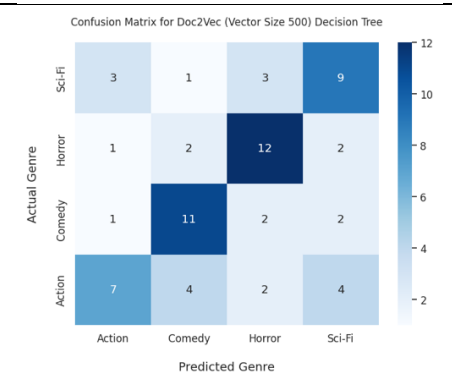
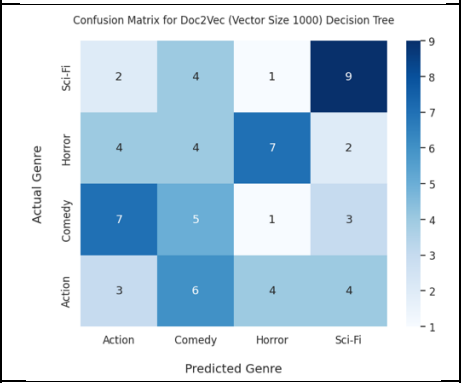
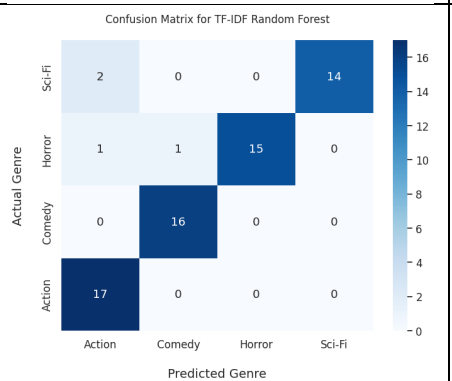
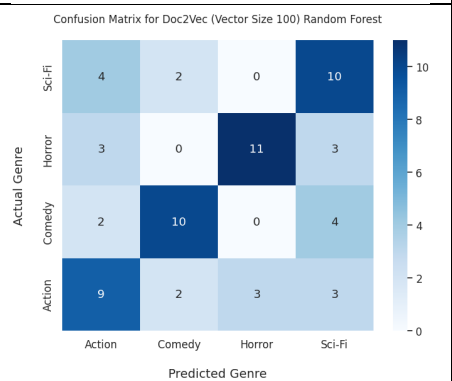
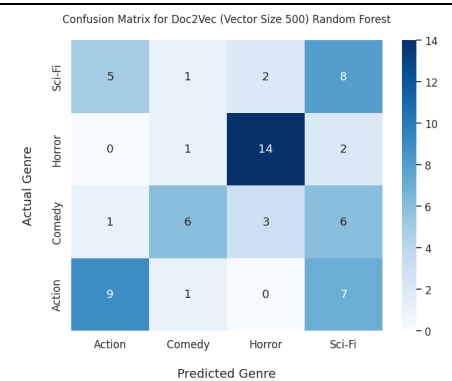
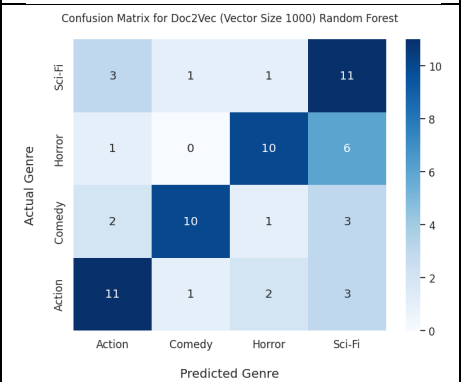
**Doc2Vec Size 500**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.49 | 0.52 | 0.50 | 33 |
| Positive | 0.48 | 0.45 | 0.47 | 33 |
| Accuracy | | | 0.48 | 66 |
| Macro Avg | 0.48 | 0.48 | 0.48 | 66 |
| Weighted Avg | 0.48 | 0.48 | 0.48 | 66 |

**Doc2Vec Size 1000**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Negative | 0.57 | 0.61 | 0.59 | 33 |
| Positive | 0.58 | 0.55 | 0.56 | 33 |
| Accuracy | | | 0.58 | 66 |
| Macro Avg | 0.58 | 0.58 | 0.58 | 66 |
| Weighted Avg | 0.58 | 0.58 | 0.58 | 66 |

## A6. Genre Classification Confusion Matrices

| Model | TF-IDF | Doc2Vec Size 100 | Doc2Vec Size 500 | Doc2Vec Size 1000 |
|---|---|---|---|---|
| **Support Vector Machine** |  |  |  |  |
| **Decision Tree** |  |  |  |  |
| **Random Forest** |  |  |  |  |

## A7. Genre Classification Performance Metrics

### Support Vector Machine

**TF-IDF**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 1.00 | 1.00 | 1.00 | 17 |
| Comedy | 1.00 | 1.00 | 1.00 | 16 |
| Horror | 1.00 | 1.00 | 1.00 | 17 |
| Sci-Fi | 1.00 | 1.00 | 1.00 | 16 |
| Accuracy | | | 1.00 | 66 |
| Macro Avg | 1.00 | 1.00 | 1.00 | 66 |
| Weighted Avg | 1.00 | 1.00 | 1.00 | 66 |

**Doc2Vec Size 100**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.93 | 0.82 | 0.87 | 17 |
| Comedy | 0.93 | 0.88 | 0.90 | 16 |
| Horror | 0.94 | 0.94 | 0.94 | 17 |
| Sci-Fi | 0.74 | 0.88 | 0.80 | 16 |
| Accuracy | | | 0.88 | 66 |
| Macro Avg | 0.89 | 0.88 | 0.88 | 66 |
| Weighted Avg | 0.89 | 0.88 | 0.88 | 66 |

**Doc2Vec Size 500**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 1.00 | 1.00 | 1.00 | 17 |
| Comedy | 0.94 | 1.00 | 0.97 | 16 |
| Horror | 1.00 | 1.00 | 1.00 | 17 |
| Sci-Fi | 1.00 | 0.94 | 0.97 | 16 |
| Accuracy | | | 0.98 | 66 |
| Macro Avg | 0.99 | 0.98 | 0.98 | 66 |
| Weighted Avg | 0.99 | 0.98 | 0.98 | 66 |

**Doc2Vec Size 1000**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 1.00 | 1.00 | 1.00 | 17 |
| Comedy | 0.94 | 1.00 | 0.97 | 16 |
| Horror | 1.00 | 1.00 | 1.00 | 17 |
| Sci-Fi | 1.00 | 0.94 | 0.97 | 16 |
| Accuracy | | | 0.98 | 66 |
| Macro Avg | 0.99 | 0.98 | 0.98 | 66 |
| Weighted Avg | 0.99 | 0.98 | 0.98 | 66 |

### Decision Tree

**TF-IDF**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.60 | 0.35 | 0.44 | 17 |
| Comedy | 0.71 | 0.62 | 0.67 | 16 |
| Horror | 0..76 | 0.76 | 0.76 | 17 |
| Sci-Fi | 0.44 | 0.69 | 0.54 | 16 |
| Accuracy | | | 0.61 | 66 |
| Macro Avg | 0.63 | 0.61 | 0.60 | 66 |
| Weighted Avg | 0.63 | 0.61 | 0.60 | 66 |

**Doc2Vec Size 100**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.73 | 0.47 | 0.57 | 17 |
| Comedy | 0.43 | 0.38 | 0.40 | 16 |
| Horror | 0.47 | 0.53 | 0.50 | 17 |
| Sci-Fi | 0.36 | 0.50 | 0.42 | 16 |
| Accuracy | | | 0.47 | 66 |
| Macro Avg | 0.50 | 0.47 | 0.47 | 66 |
| Weighted Avg | 0.50 | 0.47 | 0.48 | 66 |

**Doc2Vec Size 500**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.58 | 0.41 | 0.48 | 17 |
| Comedy | 0.61 | 0.69 | 0.65 | 16 |
| Horror | 0.63 | 0.71 | 0.67 | 17 |
| Sci-Fi | 0.53 | 0.56 | 0.55 | 16 |
| Accuracy | | | 0.59 | 66 |
| Macro Avg | 0.59 | 0.59 | 0.59 | 66 |
| Weighted Avg | 0.59 | 0.59 | 0.59 | 66 |

**Doc2Vec Size 1000**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.19 | 0.18 | 0.18 | 17 |
| Comedy | 0.26 | 0.31 | 0.29 | 16 |
| Horror | 0.54 | 0.41 | 0.47 | 17 |
| Sci-Fi | 0.50 | 0.56 | 0.53 | 16 |
| Accuracy | | | 0.36 | 66 |
| Macro Avg | 0.37 | 0.37 | 0.37 | 66 |
| Weighted Avg | 0.37 | 0.36 | 0.36 | 66 |

### Random Forest

**TF-IDF**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.85 | 1.00 | 0.92 | 17 |
| Comedy | 0.94 | 1.00 | 0.97 | 16 |
| Horror | 1.00 | 0.88 | 0.94 | 17 |
| Sci-Fi | 1.00 | 0.88 | 0.93 | 16 |
| Accuracy | | | 0.94 | 66 |
| Macro Avg | 0.95 | 0.94 | 0.94 | 66 |
| Weighted Avg | 0.95 | 0.94 | 0.94 | 66 |

**Doc2Vec Size 100**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.50 | 0.53 | 0.51 | 17 |
| Comedy | 0.71 | 0.62 | 0.67 | 16 |
| Horror | 0.79 | 0.65 | 0.71 | 17 |
| Sci-Fi | 0.50 | 0.62 | 0.56 | 16 |
| Accuracy | | | 0.61 | 66 |
| Macro Avg | 0.62 | 0.61 | 0.61 | 66 |
| Weighted Avg | 0.63 | 0.61 | 0.61 | 66 |

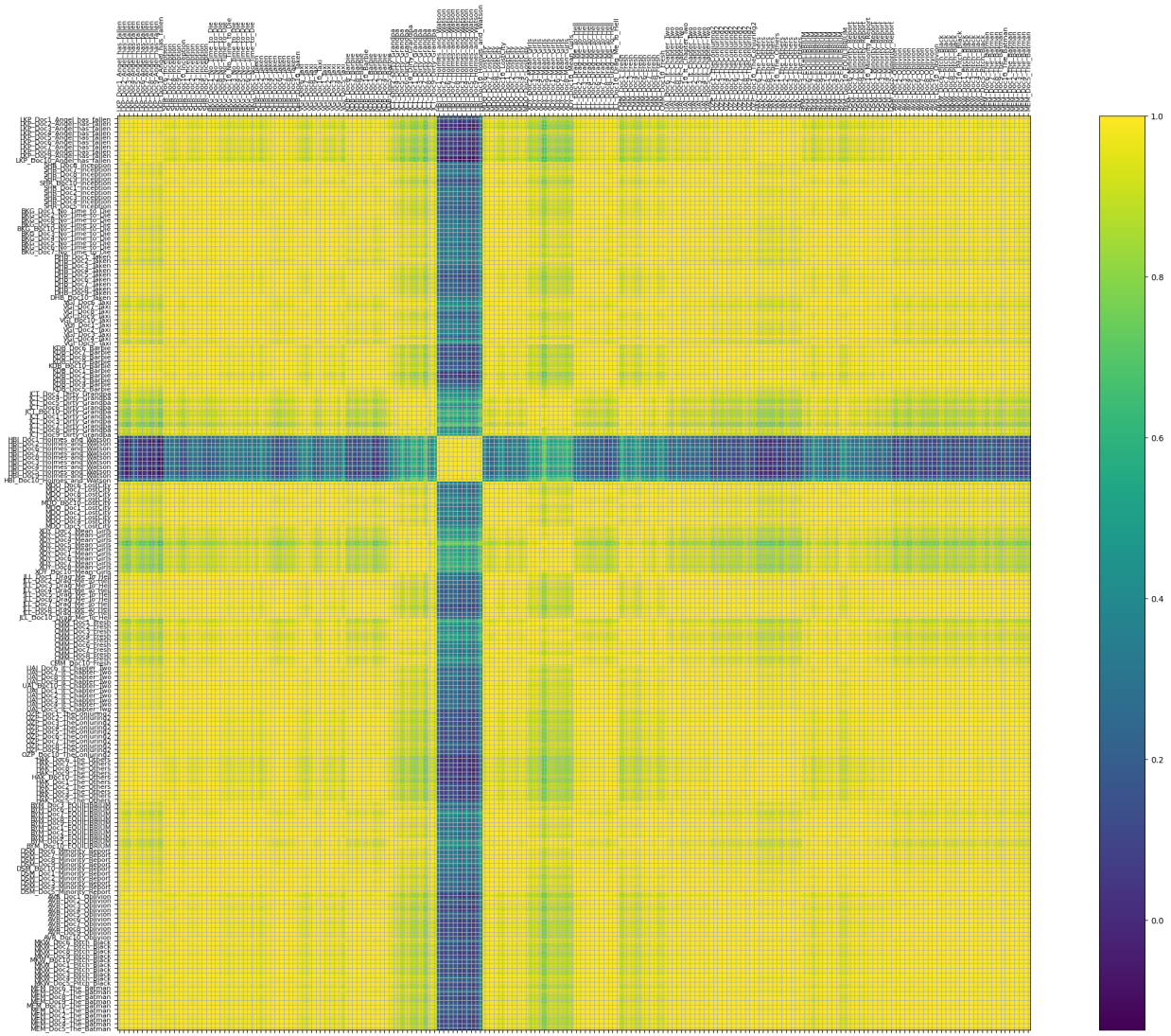
**Doc2Vec Size 500**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.60 | 0.53 | 0.56 | 17 |
| Comedy | 0.67 | 0.38 | 0.48 | 16 |
| Horror | 0.74 | 0.82 | 0.78 | 17 |
| Sci-Fi | 0.35 | 0.50 | 0.41 | 16 |
| Accuracy | | | 0.56 | 66 |
| Macro Avg | 0.59 | 0.56 | 0.56 | 66 |
| Weighted Avg | 0.59 | 0.56 | 0.56 | 66 |

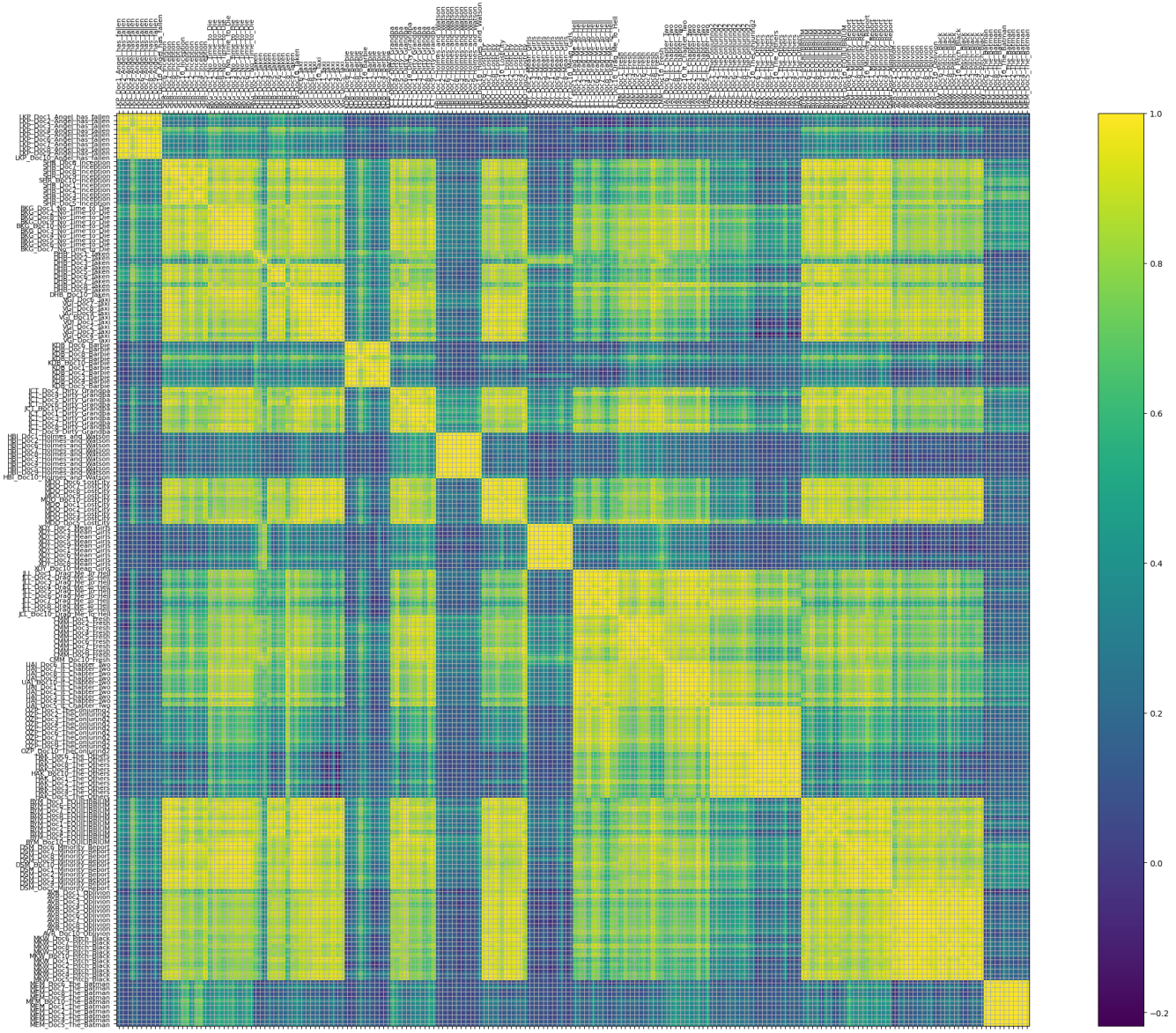
**Doc2Vec Size 1000**

| | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| Action | 0.65 | 0.65 | 0.65 | 17 |
| Comedy | 0.83 | 0.62 | 0.71 | 16 |
| Horror | 0.71 | 0.59 | 0.65 | 17 |
| Sci-Fi | 0.48 | 0.69 | 0.56 | 16 |
| Accuracy | | | 0.64 | 66 |
| Macro Avg | 0.67 | 0.64 | 0.64 | 66 |
| Weighted Avg | 0.67 | 0.64 | 0.64 | 66 |

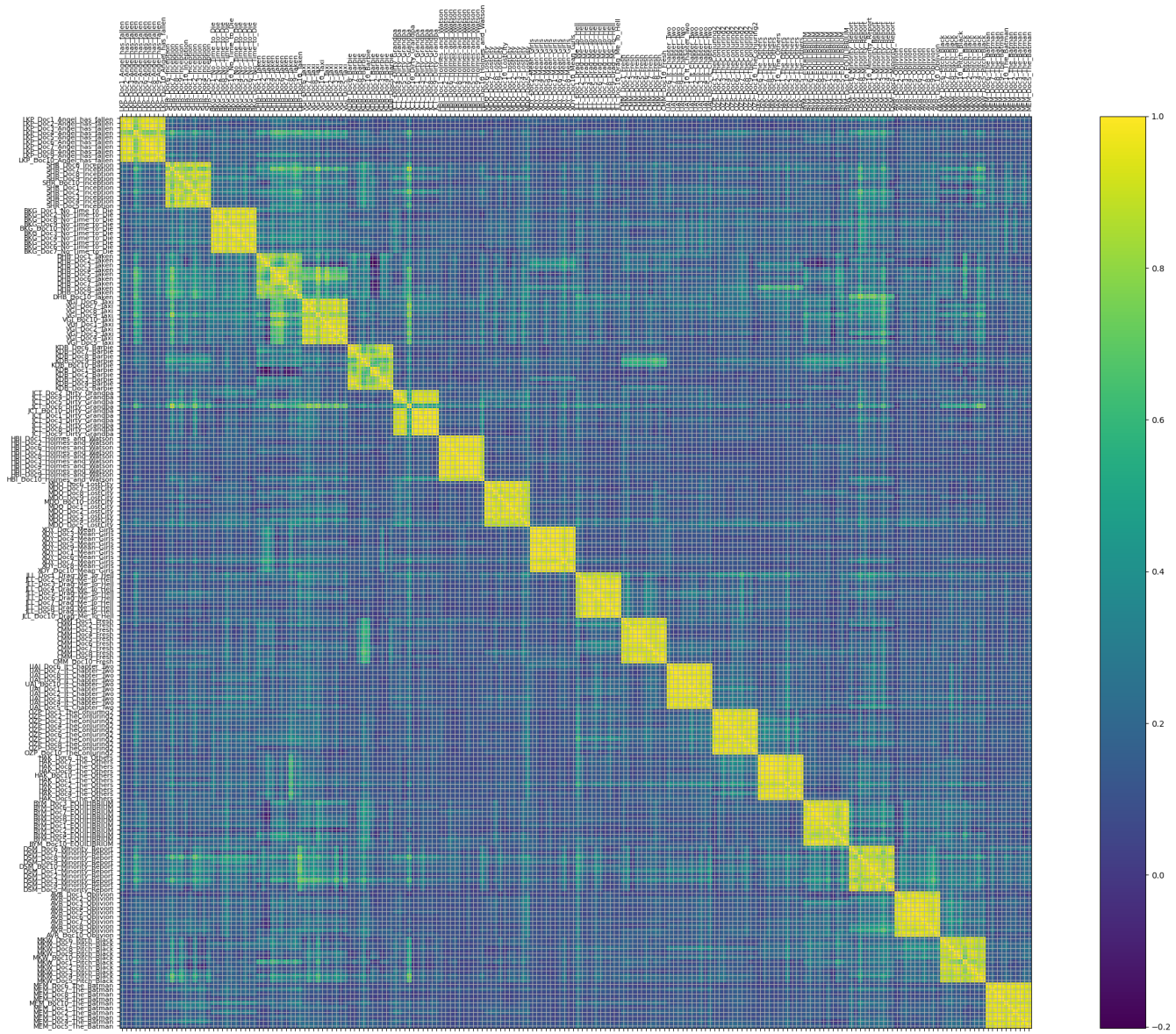
## A8. Latent Semantic Analysis (LSA) 2 Concepts, 10 Words Similarity Heatmap



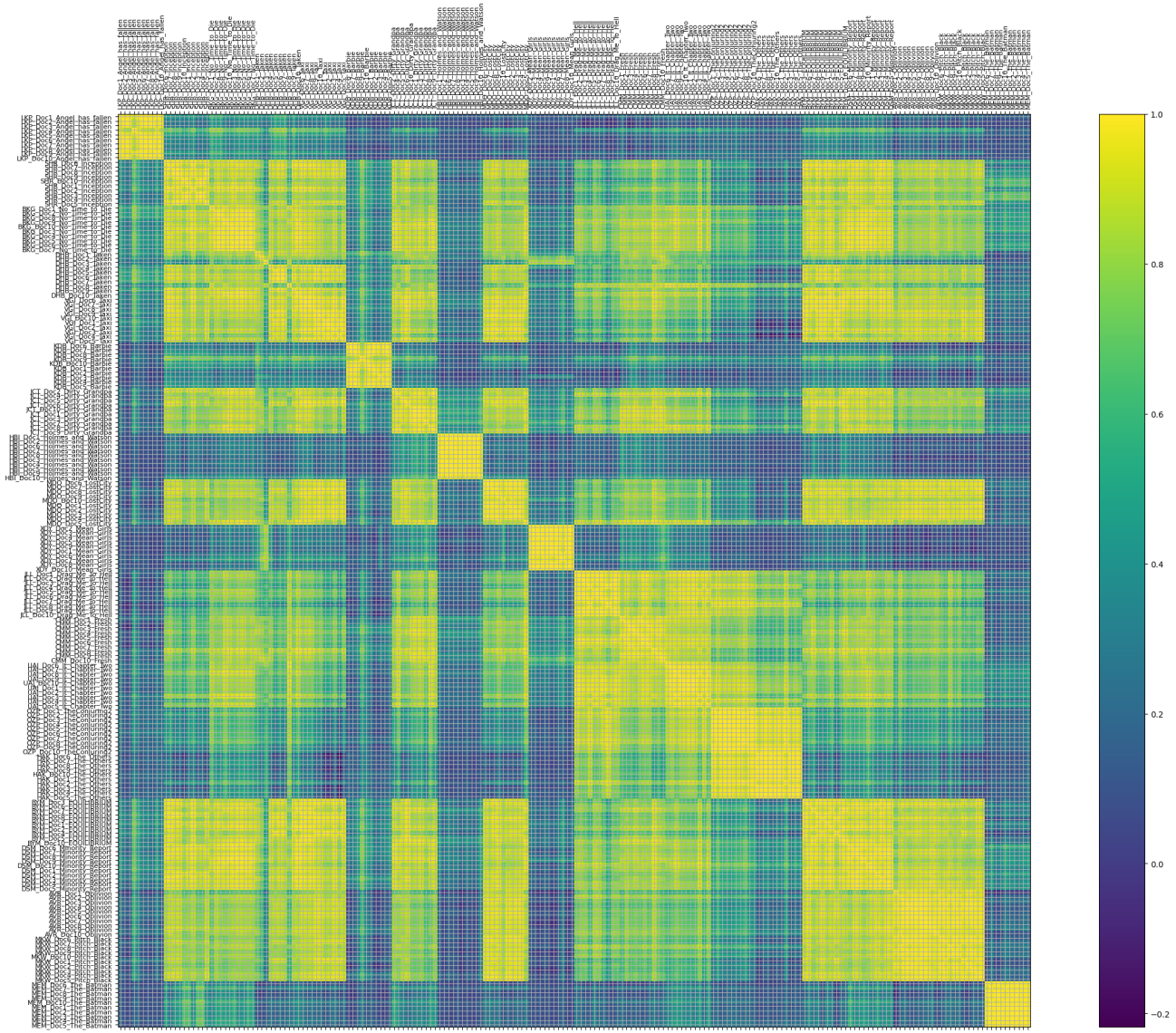
## A9. Latent Semantic Analysis (LSA) 4 Concepts, 10 Words Similarity Heatmap



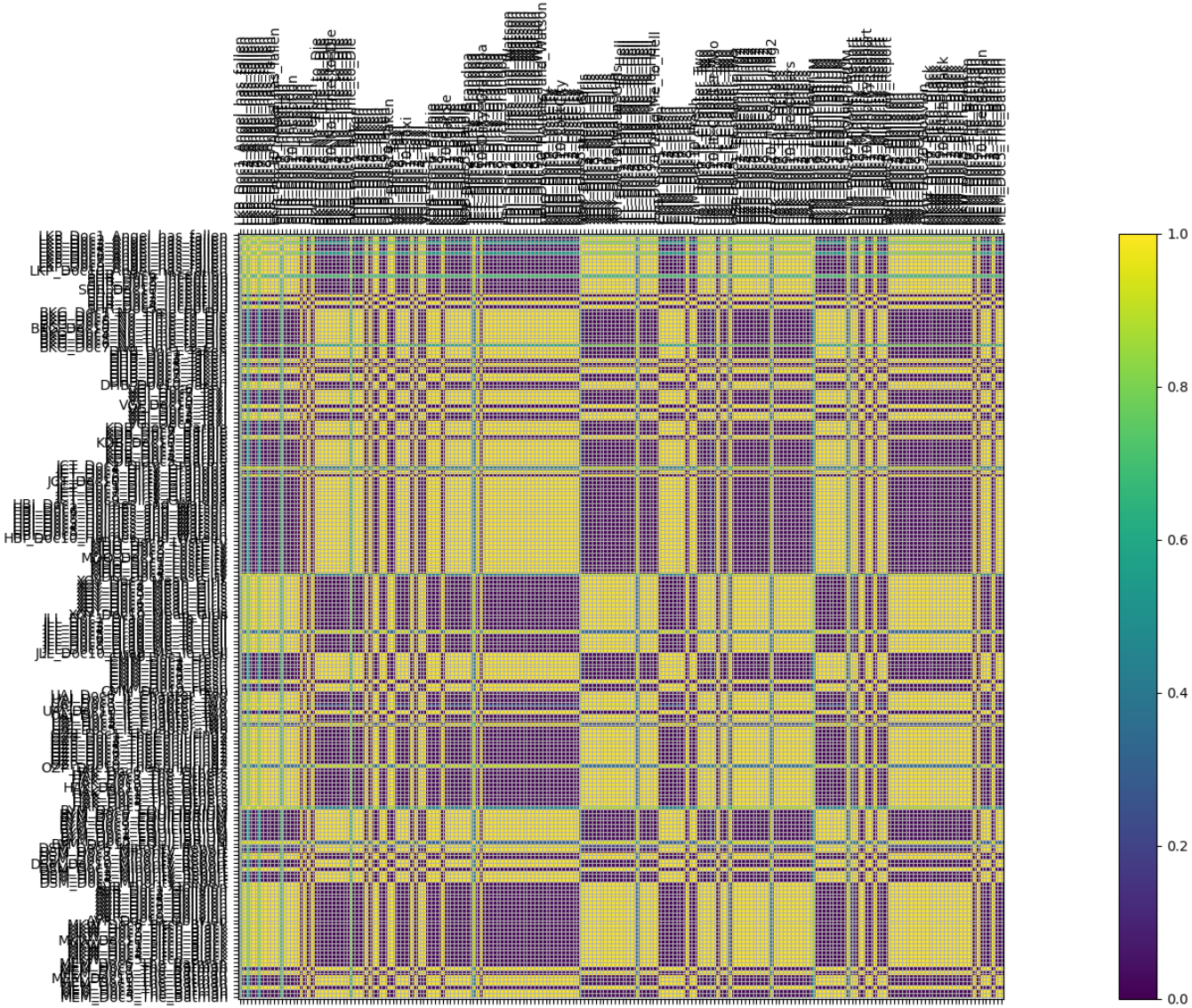
## A10. Latent Semantic Analysis (LSA) 20 Concepts, 10 Words Similarity Heatmap



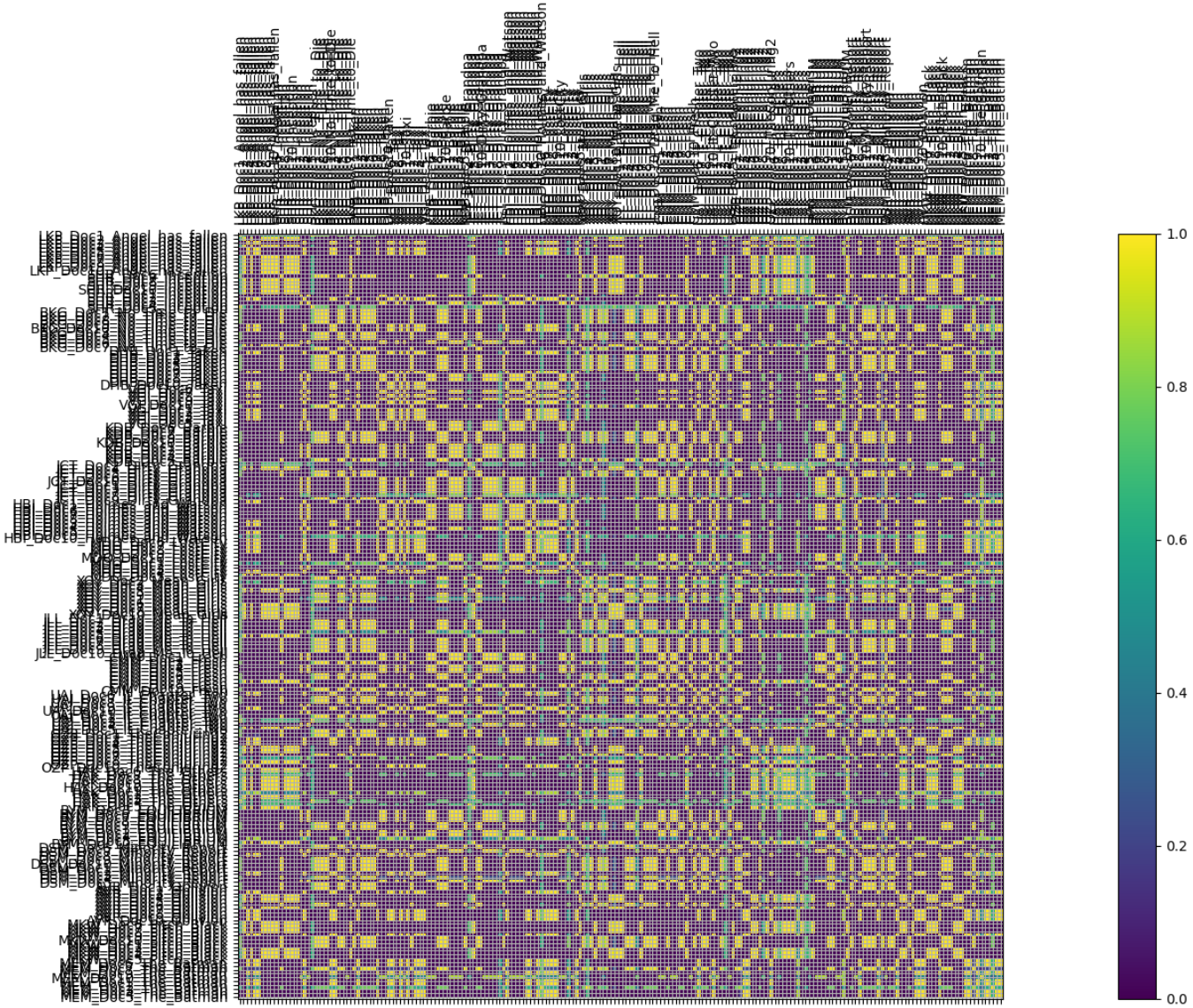
## A11. Latent Semantic Analysis (LSA) 7 Concepts, 10 Words Similarity Heatmap



**A12. Latent Dirichlet Allocation (LDA) 2 Topics, 20 Words Similarity Heatmap**



**A13. Latent Dirichlet Allocation (LDA) 4 Topics, 20 Words Similarity Heatmap**



## A14. Latent Dirichlet Allocation (LDA) 18 Topics, 20 Words Similarity Heatmap