Amazon Movie Analysis : CS 5664 Final Project

CONRAD PEREIRA, Virginia Tech, USA
DIVYA JYOTHI GADDIPATI, Virginia Tech, USA

Abstract: Generating recommendations for movie streaming platforms is becoming increasingly important. Providing a useful service results in having high user engagement which can increase revenue from more subscriptions. We explore collaborative content recommendation techniques using Scipy's Surprise library's implementations of common machine learning algorithms for content recommendations. Beyond this we will do a sentiment analysis and topic modeling of reviews, and a network analysis of the bipartite network created between reviewers and products. This will allow us to develop an understanding of the Amazon Movie and TV show data set and to better understand customer behaviour and demands. The best accuracy of the models used to generate recommendations achieved was a RMSE of 0.939 on the Amazon Movie and TV Shows dataset. With sentiment analysis, the best model achieved an accuracy of 0.78 when Word2Vec + SVM is used. We also present insights on the words that are associated with positive, negative and neutral sentiments by visualizing them in a word cloud. Streamlining the huge corpus of reviews into topics helped in gaining a better understanding the kind of content/genre that is popular among the community.

CCS Concepts: • **Social Media Analytics**; • **Machine Learning (Recommender Systems)**; • **Sentiment Analysis**; • **Topic Modeling**; • **Network Analysis**;

Additional Key Words and Phrases: Recommendation Systems, Sentiment Analysis, Topic Modeling, Network Analysis

ACM Reference Format:
Conrad Pereira and Divya Jyothi Gaddipati. 2022. Amazon Movie Analysis : CS 5664 Final Project. *ACM Trans. Graph.* 37, 4, Article 111 (August 2022), 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Systems to recommend products related to movies and TV shows are essential for user engagement for Netflix, Hulu, Amazon Prime, and other online streaming services. To improve these recommender systems we will attempt to understand popularity of different movies and TV shows, and recommend movies to users based on other movies they have liked.

The value of being able to recommend a product in general that will be entertaining or useful to a consumer is huge. Having a streaming service that keeps good movies coming based on what a user has already seen will keep the customer more entertained and more likely to continue using the service. This keeps the user happy and potentially increases the companies revenues over time as more people are happy with the product. We will use model-based collaborative filtering techniques from the surprise library to generate recommendation models and then we will test them using K-fold Cross Validation to gather the average RMSE for each model.

Sentiment Analysis and Topic modeling will provide us with a deeper understanding of how individuals rate various products. We can also determine whether a given review contains negative, positive, or neutral emotions and find the general topic of a given review to determine whether there are any major trends in negative or positive reviews we can look at to improve issues. For sentiment analysis, we'll explore two approaches: (1) lexicon-based model and (2) supervised machine learning models on processed reviews using pretrained word embedding model.

1.1 Amazon Product Review Data

The Amazon Product Review data contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014. Information including reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs) are also available.

For this project, the Movies and TV reviews (4,607,047 reviews) will be used in order to attempt to improve the recommended movies based on other movies they have liked. The dataset contains reviews (textual information), ratings of movies (endorsement/recommendations). For this work, we will not be concerned with the textual content because we are using collaborative recommendation techniques rather than content-based. The data does not contain network information directly, though a bipartite graph from users (reviewerID) to the movies watched/rated (asin - ID of the product) can be created from the data. In turn, this graphs features can be evaluated with various methods from the course, and it may provide some useful information/visuals on the most popular movies. Our minor contribution to the world of recommendation systems is the simple testing of five of the models in the Surprise library on a different data set, originally the testing the creators of this library used was on the MovieLens dataset.

1.2 Features

Listed below are each of the features from the Amazon Movie Product Review Data, and a brief description.

reviewerID	ID of the reviewer, e.g. A2SUAM1J3GNN3B
asin	ID of the product, e.g. 0000013714
reviewerName	name of the reviewer
helpful	helpfulness rating of the review, e.g. 2/3
reviewText	text of the review
overall	rating of the product
summary	summary of the review
unixReviewTime	time of the review (unix time)
reviewTime	time of the review (raw)

Authors' addresses: Conrad Pereira, conradpereira@vt.edu, Virginia Tech, P.O. Box 1212, Dublin, Ohio, USA, 43017-6221; Divya Jyothi Gaddipati, divyaj@vt.edu, Virginia Tech, P.O. Box 1212, Dublin, Ohio, USA, 43017-6221.

2 APPROACHES AND BACKGROUND/RELATED WORK

2.1 Content Recommendation

Content Recommendations will be performed with collaborative methods. These collaborative filtering techniques can determine items that a user may like based on ratings of other users. This is a great method, but can suffer from the cold start problem, meaning that we need a significant amount of users with ratings to begin using these techniques. If we did not have a data set of users, products, and ratings, we may have to use content-based recommendation techniques. These content-based approaches require information related to the product rather than using ratings. In our use case, we are looking at movies so we may use the movie scripts text to find other similar movies to provide recommendations.

We will experiment with primarily model based collaborative filtering, but will look at memory based as well. The model based methods we will use include K-Nearest Neighbor, Matrix Factorization (SVD), and random estimator from the Surprise library. We will test the performance using Root Mean Squared Error (RMSE), because it allows for easy interpretation and a quick number to reference. Human evaluation of recommendations generated may be a good option to go by as well. For instance, we could test each of the recommendation models in the wild to see how much users actually engage with the recommendations created and form metrics based on that.

A popular algorithm for Collaborative Filtering is the K Nearest Neighborhood (KNN) algorithm. As the name suggests, we find the most similar users (nearest neighbors) to a user we are making recommendations for, and suggest movies that the nearest neighbor also likes. Two users are similar if they provide somewhat similar ratings for the same movies and TV shows. A huge problem with this, as mentioned, is the cold start problem. We can't use this method if we don't have a significant amount of users with ratings already.

The SVD model we will use is yet another collaborative filtering method. A matrix where rows are the reviewrs and the column is a movie or tv show is generated. The individual cells are the integer ratings the reviewer has given each movie or tv show. Basically, SVD Matrix factorization can provide us with how much a user is aligned with some features, and how much a movie fits into this set.[1]

More advanced methods for content recommendation were not explored for this project due to time/computing constraints. An interesting method was covered in "Temporal-Contextual Recommendation in Real-Time" where they developed HRNN (Hierarchical Recurrent Neural Networks) that was trained on the Movie Lens data set and various others. With this model, the researchers achieved a RMSE of 0.846 which outperformed other state of the art models for recommendations systems on the same data sets. Parts of HRNN-meta have been deployed in production at scale for customers to use at Amazon Web Services and serves as the underlying recommender engine for thousands of websites.[2]

2.2 Sentiment Analysis of Reviews

Understanding customer's behavior and demands in relation to a company's products and services is crucial. This can help in determining how satisfied the customers are with the products/services by interpreting the feedback through product reviews. This is especially useful for streaming services to be able to retain good content and remove less popular content based on the customers' reviews. The most popular way to gain insights into customers' reviews is through performing sentiment analysis to determine whether a review is positive or negative or neutral. Although, it is very labor intensive to perform such a task manually which would require to read through the overwhelmingly increasing list of long reviews. Hence, automatic classification of the reviews is the only practical method for effective sentiment classification.

There has been a significant progress on sentiment classification on textual data. In this project, we explore the popular lexicon-based technique called Valence Aware Dictionary and Sentiment Reasoner (VADER). VADER relies on a dictionary that maps the lexical features of a word to emotion intensities or sentiment scores. Then the sentiment score of a text can be obtained by summing up the sentiment score of each word in the text. Thus, VADER is sensitive to both polarity (positive/negative) and intensity (strength) of emotion.

Machine learning for NLP has also been very popular for many years and have been successful because of the huge amount of data that is available for training the models. For these models, it is required that the data is in integer format. For this purpose, we need to first convert the text into a fixed length vector embedding in such a way that each word maps to a vector which also captures the semantic similarity and dissimilarity between words in this new vector space. Word2Vec is a common technique used for this purpose.

We leverage some of the supervised machine learning models for classifying the sentiment of the reviews like Naive Bayes, Logistic Regression and SVM. Naive Bayes is a probabilistic approach which uses the Bayes rule for computing the posterior probability to classify based on the input. As the data is continuous, we specifically use the Gaussian Naive Bayes for classifying the data. Logistic Regression (LR) is classification algorithm which uses a logistic function (sigmoid) to assign observations to a discrete set of classes. LR is a robust technique for two-class and multi-class classification. SVM is another robust supervised learning technique that uses decision boundaries (hyperplanes) to divide data into two or more groups. The goal is to obtain optimal decision boundaries that will minimize the classification error but also increase generalization ability of the model.

2.3 Topic Modeling of Reviews

Topic modeling helps in figuring out which topics create the input documents. It is an unsupervised technique that is used to identify groups of words (that together make a topic) within a collection of texts. Topics are repeating patterns of co-occurring terms in a collection. One of the methods for obtaining these topics is Latent Dirichlet Allocation (LDA). It is an iterative model that starts with a fixed number of topics where topics are represented as a probability distribution over the terms. In turn, each review is represented as a probability distribution over topics. Together, these distributions provide a sense of the ideas within the reviews. We try to analyze the kind of movies and tv shows are popular among the users.

2.4 Network Analysis

The original data contains no network information directly, a bipartite graph from users (reviewerID) to the movies watched/rated (asin - ID of the product) was created from the data. With each user id, an edge to a movie rated was created. In turn, this graphs features can be evaluated and we can arrive at relevent conclusions based, and it provides useful Gephi visuals on the more popular movies. Bipartite Networks are useful though it can be difficult to analyze in comparison to other types of networks because the reviewers can only reach each other by going to another product first, which adds an interesting element to the interpretation of it.

Various graph features of the network we created will be covered in the experimentation/analysis section. The number of nodes, edges, average degree, number of connected components, density, diameter, average shortest path, number of communities, centrality metrics, and the clustering coefficient are used to provide a basic analysis of the network.

3 EXPERIMENTATION/ANALYSIS

3.1 Content Recommendation

Now that we have prepared the Amazon Movie and TV show dataset for use with the Surprise library and have a better understanding, we can select the recommendation models we will use to generate recommendations by looking at the RMSE. The dataset was altered to only include reviewers with more than 100 ratings, which may have had an impact on the results, the dataset was also converted to a Surprise friendly dataset before use with the library. We will focus on using these models from the Surprise library below:
- K-nearest neighbors (KNN) Basic and With Means
- Baseline Only
- Matrix Factorization SVD
- Normal Predictor

After hyper parameter tuning, we ended up selecting k=15 for the KNNBasic and k=50 for KNNWithMeans, and for the baseline and normal predictor models the default settings were used as there are no parameters. The SVD model took a significant amount of time to do a gridsearch on, but we ended arriving at the following hyper parameters: n_epochs=20, lr_all=0.005, reg_all=0.4 to maximize performance for the parameters we used.

After training the models, we can now examine the performance of models using k-fold cross validation. By examining figure 1 below, we can see that the baseline only model actually had the lowest RMSE (0.939) in comparison to the other models. The SVD model (0.95) and KNN With Means (0.965) both performed almost as well. The KNN With Means, SVD, and BaselineOnly models could all be used as our optimal model as the final results are somewhat similar. Further testing may be required to make a choice with statistical significance.

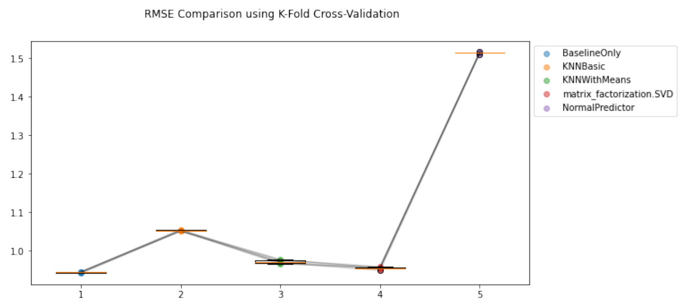


Fig. 1. RMSE Score Comparisons for the recommendation models.

Model	RMSE
KNN Basic	1.04
KNN With Means	0.965
BaselineOnly	0.939
SVD (Matrix Factorization)	0.95
Normal Predictor	1.51

Table 1. Comparison of performance of Surprise library models on the Amazon Movie and TV show data

3.2 Sentiment Analysis of Reviews

3.2.1 Data preprocessing. The dataset contains full review and summary of the review which will be used to perform the sentiment analysis. Along with the full review and summary of the review, the dataset also contains the overall rating (1 - 5) provided by the customer as well. For the purpose of the experiments and to analyse the performance of the models, the overall rating is used to assign a ground-truth sentiment label. The ratings of 1 and 2 are labelled as 'negative', rating of 3 is labelled as 'neutral' and ratings of 4 and 5 are labelled as 'positive'.

Sentiment type	Number of samples	Percentage
Positive (UR = 4,5)	1289602	75%
Negative (UR = 1,2)	206629	12%
Neutral (UR = 3)	201302	11%

Table 2. Number of samples across each sentiment based on user rating (UR)

Text preprocessing is an important step for sentiment analysis. As user-generated content is typically unstructured, it should first be cleaned and normalized before feeding the data to any model. The following steps are performed for text normalization:
- Removing any special characters and punctuation.
- Removing urls.
- Removing characters like newline, tabs etc.
- Removing stopwords that add no significant value. NLTK stopwords corpus is used for this purpose. However, negative words like 'not', 'no', 'cannot' and similar such words are often significant in determining the sentiment. Hence, these are excluded from the stopwords list.

3.2.2 Sentiment Analysis using VADER. The text preprocessing is applied all full-reviews which are then used for performing sentiment analysis. Since, summary of the review is already a short text and assuming it doesn't have any trivial words, no text preprocessing is performed. To understand the effect of the summary and the full text on the sentiment, the following experiments are performed.

- Sentiment Analysis on full-review alone
- Sentiment Analysis on summary alone
- Sentiment Analysis on full-review and summary combined

To show the performance, we present the weighted average accuracy, precision, recall and F1-scores. The performance of each of the above variations is shown in Table 3 and the corresponding confusion matrices [5] are shown in Figure 3. The performance was highest when the full review combined with summary is fed into the model. This is evident as the summary when combined with the full review would tend to enhance the actual sentiment of the text. It is interesting to note the results in case of summary-alone are heavily skewed towards neutral sentiment with. This proves it can often be difficult to determine the sentiment of very short reviews and many positive and negative reviews seemed to be misclassified as neutral. For the longer texts, the results also show that it is often easier to classify positive reviews than negative or neutral reviews. This could also be because the dataset is imbalanced with mostly positive sentiment samples.

Text type	Accuracy	Precision	Recall	F1-score
Full-review alone	0.68	0.70	0.68	0.69
Summary alone	0.39	0.77	0.39	0.46
Full-review + summary	0.71	0.71	0.71	0.71

Table 3. Comparison of performance of VADER on different texts

To understand what are the typical words in reviews that influence the sentiment, a word cloud is generated for each of the sentiments individually using the WordCloud library [4]. These are shown in Figures 5 - 7.

- **Word cloud generated from full review:** There a quite a few overlappings between positive and neutral words. We can see the words like favorite, charming, wonderful, christmas, good, best are coupled with positive sentiment as expected. It is interesting to see that the word 'classic' is in all the three word clouds. It is surprising to see depression and nostalgia in the negative word cloud.
- **Word cloud generated from summary alone:** Evidently, more positive words like love, outstanding, brilliant, , fantastic etc., appear in the positive word cloud. It is interesting to note that the model was able to relate 'great' and 'best' as positive and 'good' as neutral. More negative words like odd, problems, terrible, scam, negative, etc., appear in the negative word cloud.
- **Word cloud generated from full review + summary:** The words in each sentiment cloud seem to be a combination of the above two word clouds. The word movie occurs in almost all word clouds. This shows that content-specific stopwords should be removed as these are commonly occurring (movie, dvd in this case) and provide little value to determine the sentiment.

3.2.3 Sentiment Analysis using Supervised Machine Learning. The supervised machine learning models require the text to be encoded in integer format. This is done using the pretrained Word2Vec model available in the spaCy library. Using this pretrained embedding model, every review is converted into a 300-dimensional word vector which will be fed to the model for training. This took computationally expensive due to the huge corpus of reviews. We use the full review and summary combined text for training the ML models.

The data is split in such a way that 80% is used for training and 20% for testing with stratification which led to 1358026 training samples and 339507 samples. The performance metrics are shown in Table 4 and the corresponding confusion matrices in Figure 4. From the results, we can observe that ML models, specifically Logistic regression and SVM have superior performance in classifying positive sentiment. This could be because of the huge amount of training data. However, they too struggle with classifying neutral and negative sentiments, whereas VADER seemed to be better at classifying negative sentiment than the ML models. This shows the imbalance in the data hugely affects the performance of the models. Hence, there is a need for obtaining more data which is better representative of all the classes.

Model	Accuracy	Precision	Recall	F1-score
Naive Bayes	0.49	0.71	0.49	0.55
Logistic Regression	0.78	0.72	0.78	0.72
SVM	0.78	0.73	0.78	0.71

Table 4. Comparison of performance of ML models

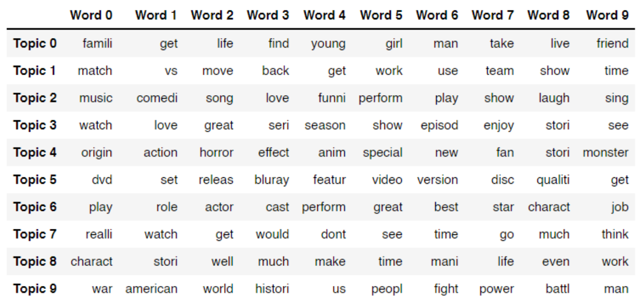
As the user reviews are often unstructured and might contain words that have not been used for training the word2vec model, the word embedding generated for such unseen words could be useless. In such cases, it would be better to perform training on the word embedding model as well to obtain better representation of all the words.

3.3 Topic Modeling of Reviews

The full text combined with summary is used to generate the topics using LDA. This forms the basis of the bag-of-words corpus for the LDA model. Prior to generating the corpus, Porter stemming is performed to reduce the words to their root/base word. The models gives the top 10 terms for each of the 10 topics as shown in Figure 2.

We can see the model generated some intelligible topics. It is interesting to see how the model accurately captures genres like comedy (Topic 2) and horror (Topic 4) and historical (Topic 9), Topic 3 seems to be referring to a TV show, Topic 5 seems to be referring to the medium of film. We can see some of the topics also refer to the movie's story and actors' performance.

This analysis is helpful in understanding the factors in movies that users are typically interested in. In terms of genre, comedy, horror and American history seem to be most popular among the users. It is also interesting to see that users also attentive and give equal importance to the kind of story and the performance of the actors.



	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9
Topic 0	famili	get	life	find	young	girl	man	take	live	friend
Topic 1	match	vs	move	back	get	work	use	team	show	time
Topic 2	music	comedi	song	love	funni	perform	play	show	laugh	sing
Topic 3	watch	love	great	seri	season	show	episod	enjoy	stori	see
Topic 4	origin	action	horror	effect	anim	special	new	fan	stori	monster
Topic 5	dvd	set	releas	bluray	featur	video	version	disc	qualiti	get
Topic 6	play	role	actor	cast	perform	great	best	star	charact	job
Topic 7	realli	watch	get	would	dont	see	time	go	much	think
Topic 8	charact	stori	well	much	make	time	mani	life	even	work
Topic 9	war	american	world	histori	us	peopl	fight	power	battl	man

Fig. 2. Dominant words in each topic

3.4 Network Analysis

The graph contains **44291 nodes, 367,867 edges**, and has an **average degree of approximately 8.305**. In this use case the nodes are representative of reviewers in the network and their edges products they have reviewed related to Movies and TV Shows from Amazon. The **graph has a diameter of 8**, meaning that the longest shortest path is 8. The diameter of a graph is the longest distance between any two nodes. The **average shortest path is 3.876.** The average shortest path is the average number of paths along the shortest paths for all possible pairs of network nodes, or people in this context. This is a dense bipartite network because between reviewers and products we must only travel an average of 3.876 nodes for the shortest path between reviewers in the network. There is a density of 0.0003, because there is 0 probability of a reviewer being connected to another reviewer, so this measurement makes sense that it is very low. There is a single connected component and a total of 9 communities, and a modularity of 0.332. The modularity measures the strength of the separation between communities, and with the number being relatively low, there is not much significant separation between communities found in this network. The 'giant connected component' for this graph is the same as the whole graph, because there is a single connected component. Therefore it also has the same clustering coefficient as the whole network. Having a single connected component means that from any person in the network we can reach any other person in the network. (Though it would have to go through a product to reach another reviewer) If there were more than 1 connected component, then some nodes are not connected by a path.

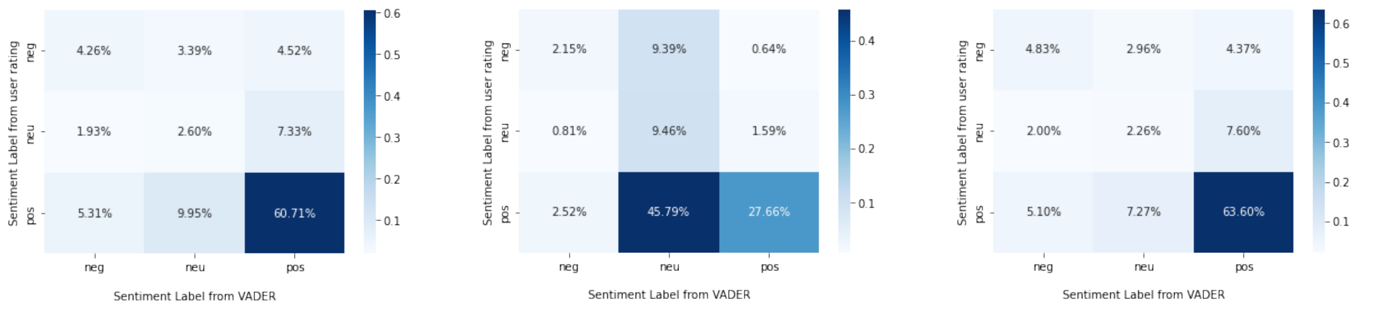
The clustering coefficient average for the Amazon Reviewer Network data is 0, meaning this network's connections are not dense at all. A single node (reviewer) has 0 connections with other reviewers, which makes sense because we have a bipartite network. On the other hand if we had a high proportion of nodes/neighbors that are connected it will result in a high clustering coefficient, and a low proportion will result in a low clustering coefficient. The clustering coefficient is a property of nodes within the network, and it demonstrates the level at which the network is connected with the value closer to 1 meaning completely connected and 0 being poorly connected. In our context, we don't have the property that most social media networks have, where people generally have friends of their friend also being their friend. This would result in sets of people among which many edges exist, more so than if connections were made at random between each person in the network. Below is the distribution of clustering coefficients for the Amazon Reviewer-Product network data, we can see that data is not connected with the average clustering coefficient being exactly 0, and all of the nodes being this value.

In figure 8, we developed a visualization with Gephi to understand the bipartite network from reviewers to products, we can see color coded communities within the network. The larger nodes correspond to nodes with higher degree, meaning that these are people with more reviews of products. These id's of the larger nodes can provide us with information on who is the most active or 'important' people in this network. In the context of movie content recommendations this is not immediately as useful as it would be in a social media network, but it allows us to develop a quick, visual understanding of the data we are working with. This visualization could also provide us with insights to provide to other teams working in a movie streaming company to have an idea of who the top reviewers are on their platform.

4 CONCLUSIONS AND FUTURE WORK

The economic value to movie streaming companies for great recommender systems is clear. To sum up our contributions, we tested the Surprise libary's implementations of common algorithms for recommending with explicit ratings including matrix factorization SVD, KNN with Means/Basic, random models, and their baseline model with the Amazon Movie and TV show dataset. Basic grid searches were done for hyperparameter tuning to find the best performing models for content recommendation with respect to the root mean squared error. The best model ended up being the baseline model (0.939 RMSE), by a small margin. This result was very interesting because, the Surprise library's testing on the Movie Len's data sets (1M and 100,000), the baseline model was one of the lower performing models, and the SVD was the best performing model.[3] Given additional time, we may build on top of Surprise library's implementation with our own algorithms to make a greater contribution to the recommendation system field of machine learning. We may also perform more exhaustive hyperparameter tuning, as we were very limited by computational resources.

Sentiment analysis helped to easily go through the huge corpus of reviews and classify them into positive, negative and neutral sentiments. We explore lexicon based and ML based models for this purpose. Among all the models SVM performed better, although upon looking at individual classes, VADER seemed to have better classification ability for neutral and negative sentiments as well. Classifying positive sentiment was easiest for almost all models. We were also able gain insights like how long/short reviews effect in determining the sentiment and that positive sentiment is easier to classify than negative or neutral sentiments. Generating word clouds helped in gaining a deeper understanding of the common words that correspond to a particular sentiment. We also observed the effect of imbalance in the data on the models' performance. It is also important to note that there might be instances in the user-generated data where new words or phrases are introduced which should be taken into account when performing the analysis.

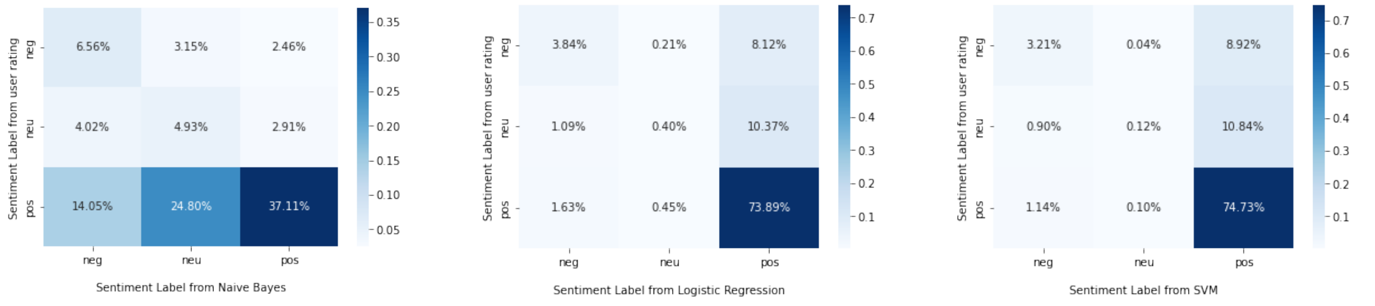


(a) Full review alone

(b) Summary alone

(c) Full review + summary

Fig. 3. Confusion matrix for VADER

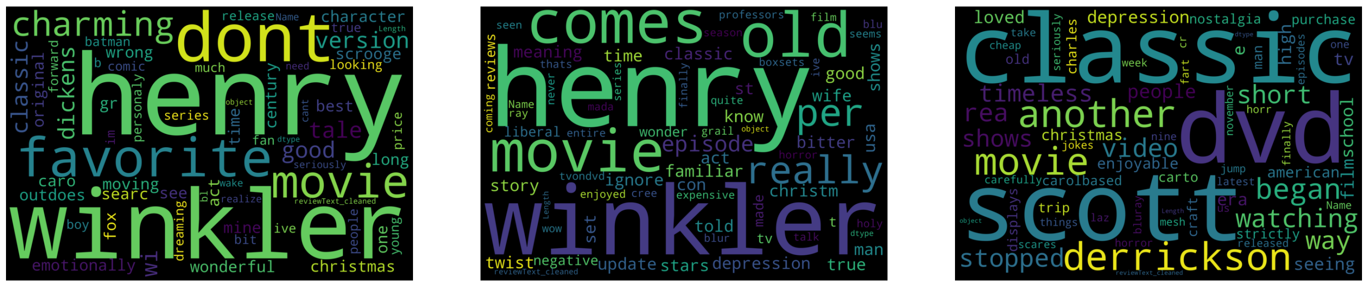


(a) Naive Bayes

(b) Logistic Regression

(c) SVM

Fig. 4. Confusion matrix for ML models trained on full review + summary



(a) Positive sentiment words

(b) Neutral sentiment words

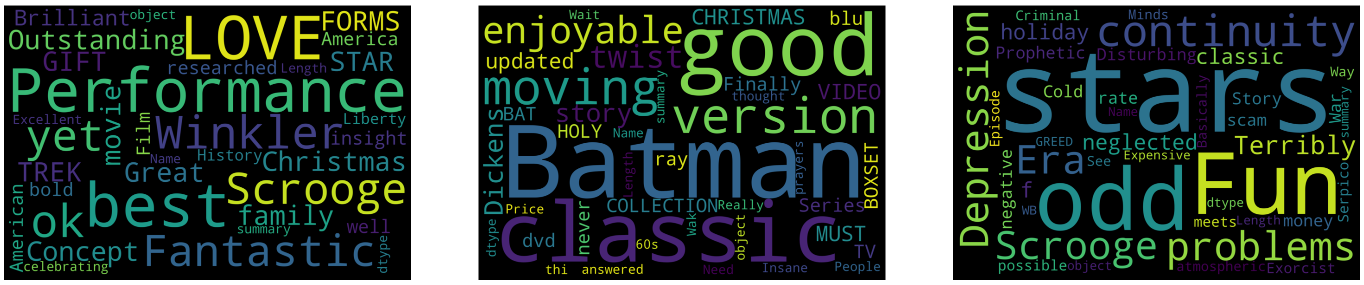
(c) Negative sentiment words

Fig. 5. Word cloud generated from full review classified using VADER

Given additional time, we could have explored training a custom word embedding model on the available dataset for obtaining a better word vector representation. For overcoming the problem of imbalanced dataset, we could explore data augmentation techniques, or collect more data for the minority sentiment classes.

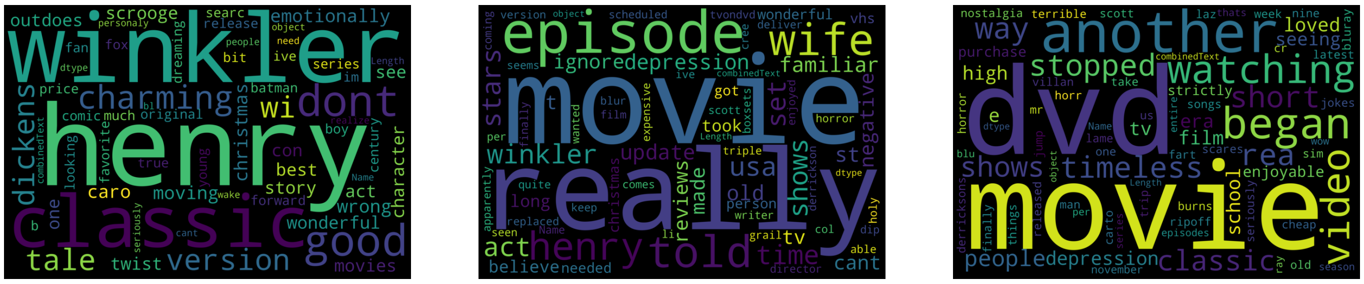
From topic modeling using LDA, we were able to gain more insights into the kind of genres that are popular. Through the dominant words in each topic, we could get a sense of the story that is related to a particular genre. As the actual movie titles were not available through the dataset, we couldn't perform analysis like finding the dominant topic for a particular genre movie. More analysis could be done in the future by combining reviews from various sources to get better insights into what users majorly care about when they look for movie or tv show recommendations.

After gaining an understanding of the bipartite network generated from reviewers to products, we also were able to understand more about the data set as a whole. Using Gephi to visualize the data can give non-technical audiences a quick glance at the most frequent reviewers/products being purchased. The graph is not even remotely dense when looking at the density metric which is expected from a bipartite network, where we only have edges going



(a) Positive sentiment words (b) Neutral sentiment words (c) Negative sentiment words

Fig. 6. Word cloud generated from summary classified using VADER



(a) Positive sentiment words (b) Neutral sentiment words (c) Negative sentiment words

Fig. 7. Word cloud generated from full review + summary classified using VADER

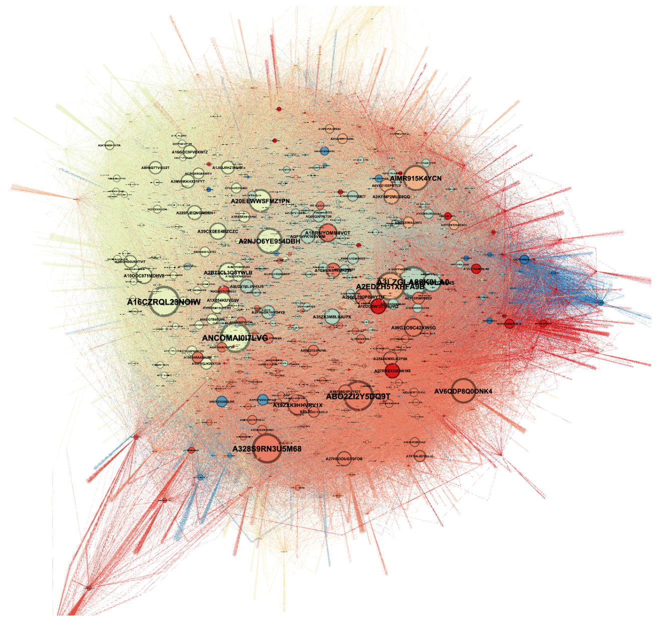


Fig. 8. Visualization of bipartite network from reviewers to products created with Gephi.

from a reviewer to a product. Aside from that the network is well connected with the average shortest being only 3.876, and having an average degree of 8.305.

5 REFERENCES

1 https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26
2 https://assets.amazon.science/96/71/d1f25754497681133c7aa2b7eb05/temporal-contextual-recommendation-in-real-time.pdf
3 http://surpriselib.com/
4 https://medium.com/analytics-vidhya/text-analysis-of-amazon-customer-reviews-b4fcf0663216
5 https://www.stackvidhya.com/plot-confusion-matrix-in-python-and-why/