Jokes Recommendation System

LAKSHYA
2021262
lakshya21262@iiitd.ac.in

NITISH KUMAR
MT23129
nitish23129@iiitd.ac.in

Sumit Bhagat
MT22302
sumit22302@iiitd.ac.in

AHMED HANOON
2021006
ahmed21006@iiitd.ac.in

ZUBAIDA FATIMA
2021221
zubaida21221@iiitd.ac.in

Abstract

Individuals have varying senses of humour, and it can be challenging to propose a novel system that can rate jokes. The data set is pre-processed using Natural Language Processing (NLP) techniques and trained on several models to get the best fit model. Average R-sqaured values of the models, obtained via K-folds validation, are compared and SVR with regularization parameter as two was selected. Finally, the system was compared with existing models and the complexity of the proposed model is analysed.

1 Introduction

Nowadays, thousands of software for good jokes personalised for different users based on artificial intelligence exist. Even smart assistants like Siri, Alexa and Google Assistant recommend jokes to people. A joke recommendation system would give an average rating for the jokes given to it. The data set to be used would be the Dataset-1 from the Jester Dataset. This data will be pre-processed using NLP techniques such as stopword removal, lemmitization and feature extraction specific to the data set. Using this data a regression model is chosen to predict ratings for jokes given to it. Lastly, this code was compared with existing systems to analyse the shortcomings and the complexity of the model is studied.

2 Existing Analysis

An analysis was done on the Jester data set as well as existing joke recommendation systems, and their performance was compared.

2.1 Data set Analysis

An analysis was done by reviewing research papers on the type of data set and how it has been pre-processed.

The data set contains 4.1 million anonymous ratings from 73,421 users on 100 jokes collected between April 1999 - May 2003. The first column contains the number of jokes rated by a user. The rating ranges from -10.00 to +10.00, with 99 representing an unrated joke.

The data set contains a lot of missing values, making the data set unclean. Due to this, pre-processing the data is essential here.

2.1.1 Data Pre-Processing

1. As part of pre-processing from [1], we do the following:
   - Jokes with + 7.00 to + 10.00 are marked as 1.
   - Jokes with ratings ranging from - 10.00 to + 6.00 are marked as zero
   - Jokes with ratings of 99 are replaced with -1 since they mean that the jokes have no rating

   This is the procedure followed in the paper, but it is flawed since ratings between +6.00 and +7.00 have not been assigned any values. This has been rectified in [2] by assigning all values below +7.00 (exclusive) as 0.
2. The paper by Costa, Silva, Antunes and Ribeiro, [3], which classifies jokes as recommendable or not, was reviewed. The data set has three parts, but this experiment uses the first part for testing since it contains a diverse collection of ratings. For classification, the data is interpreted as follows:
   - Jokes classified on average above 0.00 - ’recommendable’.
   - Jokes classified on average below 0.00 - ’non-recommendable’.

   The data was split into training and testing data as two equal disjoint sets. The training data was used to train the model, and testing data was used to evaluate the performance.

   The text jokes were indexed with a bag of terms occurring in it. After this, feature reduction was applied to reduce the feature space. This reduces the size of the data set and also removes unwanted words such as articles, prepositions, etc. The words were also represented all in one case.
3. Pre-processing for the Eigentaste [4][5] algorithm was also taken into account. There are a few variations of the data set, and this paper uses ’Dataset 3’, an updated version of their previous data set. It describes the jokes 7, 8, 13, 15, 16, 17, 18, 19 as the gauge set. A gauge set contains the jokes that were rated by everyone, so it can be used for testing and comparison of different recommender models.

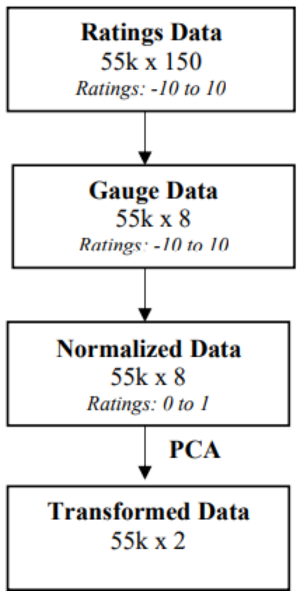

Figure 1: Flow chart demonstrating pre-processing from Eigentaste

From the raw data, the gauge data is separated and normalized by subtracting from the mean and dividing by the standard deviation. Since only the gauge set was used, there are no null values. Then, PCA was performed on the data with two principal components for ease of visualization. After this, the data is passed to the model.

Conclusion: It can be observed that out of all the pre-processing models discussed, the last one proposed by Eigenstate would be a good match since the rating is a real number instead of a binary value. This could improve the performance of the model by taking into account a range of ratings. The first pre-processing technique would be useful if the joke had to be classified as only good or bad. It would still work in a recommendation system, but the data set would not be diverse.
It can be noted that the first technique would make the data easier to work with, but with the last technique, the data set would be varied and might lead to a better prediction.

2.2 Analysis on existing models

An analysis of the existing joke recommendation models.

1. **Collaborative Filtering using Eigentaste** [3]

   Eigentaste is a faster algorithm *(O(1))* than regular nearest-neighbour techniques *(O(n))*. The fundamental idea behind Collaborative Filtering is that if a group of users rate certain items similarly, they share similar tastes, and hence will rate other items similarly. Most CF algorithms uses *user-selected queries* while Eigentaste uses *universal queries*. The advantage being that the subset of the ratings matrix containing the gauge set items is dense. It also permits the system to collect immediate feedback on all recommended items. Eigentaste also captures user ratings on a continuous rating scale.

   As part of the algorithm, the data is **normalized**, a **pearson correlation matrix** is obtained, **PCA** is performed on the data and finally **recursive rectangular clustering** is implemented on the data. Each cell is treated as a cluster of neighbors in the eigenplane. For each cluster, the mean for each non-gauge is computed based on number of users who rated that item. Sorting the non-gauge items in decreasing order of mean ratings yields a lookup table of recommendations for that cluster.

   Along with Eigentaste the study[3] also uses **POP Algorithm** and **Nearest Neighbour Algorithm**. The result are analyzed using **Normalized Mean Average Error** and is given below

Algorithm	Accuracy (NMAE)	Offline	Online	Online time per user
POP	0.203	$O(nm)$	$O(1)$	-
1-NN	0.237	-	$O(nk)$	350 msec
80-NN	0.187	-	$O(nk)$	350 msec
Eigentaste	0.187	$O(k^2n)$	$O(k)$	3.2 msec

Figure 2: Results for Eigentaste

2. **Collaborative Filtering with Mean Absolute Error, Recall and Rating Coverage** [6]

   The study by Zaier, Godin and Faucher analyzes the performance of different recommendation techniques based on different datasets including Jester Joke dataset. It was noticed that the Jester dataset do not follow a power law distribution.

   The collaborative filtering approach recommends items to a given user based on the ratings of other neighbours. Once a neighbourhood of users is formed, the rating dataset is represented using a matrix of users and items. Then, a coefficient correlation matrix for the relationship between all users is obtained. The predictions are then made for each user for each item. For the test dataset, a subset of neighbours with **20** percent of users with the closest similarity and **20** percent farthest is chosen.

   The prediction accuracy for Jester dataset showed **substantial advantages** with these methods.

3. **Bernoulli Restricted Boltzmann Machine based Recommendation System**

   RBM is an energy based graphical DL model. Gibbs Sampling is done for training the RBM following a 20 step contrastive divergence as a learning algorithm.[2]

The dataset is split into **80** percent Training Set and **20** percent Validation Set. Once the recommended ratings are obtained, they are combined to form the D1 dataset. The original values of the unrated jokes are combined to form the D2 dataset.

For Joke-Reader segmentation, K-Means clustering model is developed. The optimum value of K is obtained by the Elbow Method. No appropriate elbow value is found for D2 and D1 was chosen as Training for K-Means clustering. The trained model is then used for Dataset D2.[1]

The performance of K-Means is analyzed by visualizing the clusters using line charts. The result (using MAE) was a **0.1974** Training Loss and **0.1786** Validation Loss.

2.3 Analysis on Existing Code Repositories

An analysis on the existing code repositories. In particular, we will be analyzing repository by Abhijeet A. on Jester Recommendation Dataset [7]

First, the jokes are extracted from HTML files and inserted into SQL followed by the Ratings. The data undergoes Pre-Processing. It is normalized around zero mean. Pearson correlation is used for finding similarity.

For User-User Collaborative Filtering

1. The dataframe is then split into two, those who rated all the jokes and those who did not.
2. A random active user is selected and similarity is found between them and other complete rated users. To find the neighbours, a threshold is kept and 30 neighbours are randomly sampled.
3. A recommendation columns is created for those jokes not rated by active user. Using all this data, the highest recommended unseen joke is assigned to the user.

For Content based Joke recommendation using Topic modelling approach.

1. The raw joke data is cleaned. Common words are filtered out as it does not imply any category.
2. Lemmetization is performed over the entire joke data. Each word in these jokes are then split and assigned their role in the sentence, Part of Speech tagging.
3. Topic Modelling is performed using LDA, after which column clusters containing the probability of belonging to each topic is obtained. User ratings are normalized between -1 and 1 using Robust Scaling.
4. Similiar to User-User CF, an active user is selected and a model is trained to recommend jokes to them.

Finally Mean Squared Error is used to calculate the prediction error.

Conclusion: It can be observed that models which provide satisfactory results are collaborative filtering with Neighbours. For instance, the predictions made by Eigentaste have a normalized mean absolute error (NMAE) value of 0.187 and an accuracy as good as that of 80-NN.

3 EDA

Exploratory Data Analysis (EDA) is the most important step in building any machine learning model. It empowers developers to conduct data visualization, explore data, summarize data, and uncover valuable insights from the dataset. EDA plays a critical role in understanding the data's characteristics and patterns, making informed decisions regarding data preprocessing, and ultimately improving the performance and reliability of machine learning models

3.1 Data Cleaning

Our dataset comprises 73,418 users who have rated 100 jokes. Not all users have rated all the jokes, which is why there are numerous null values present in our dataset, representing 99. Additionally, since the given jokes contain many stopwords, we need to remove them and lemmatize each word to gain a better understanding of the jokes.

Jokes Data Cleaning : In this step, we remove non-alphanumeric characters and HTML tags from the jokes. Then, we eliminate stop words and tokenize each joke, followed by lemmatizing the jokes

However in last step we are having two choices to do stemming or lemmatizing of Jokes, Choosing lemmetization over stemming because of, Lemmatization considers the context and meaning of words, so it often retains the base form of a word. Stemming, on the other hand, can sometimes reduce words to a common root that may not make sense in the context of the sentence

User-Rating Data Cleaning : Here is the plot showing how many null values present in our dataset.

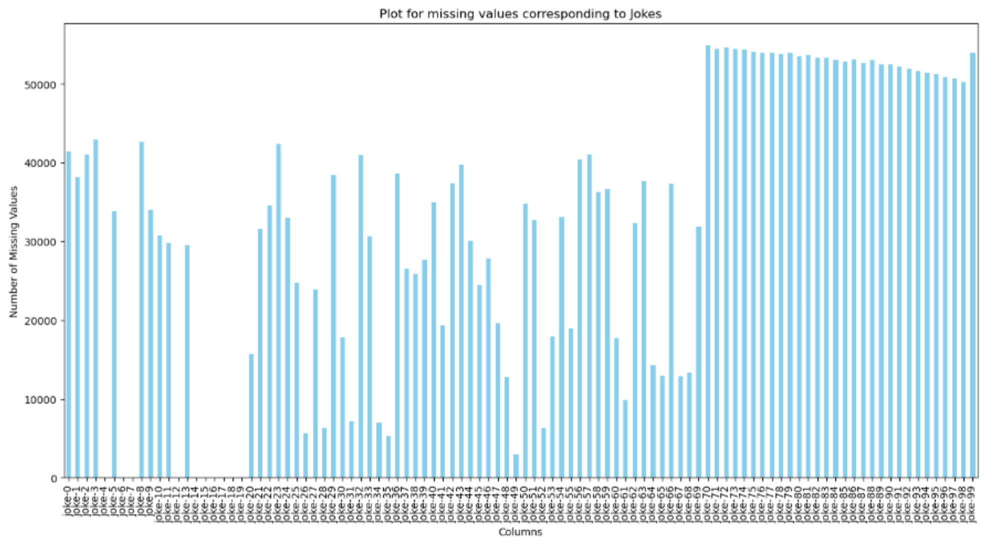

Figure 3: Count of missing values

Analyzing data with this many null values can mislead our data exploration task. Null values, representing missing or incomplete information, can distort statistical summaries, distribution analysis, and correlations within the dataset. Moreover, they can introduce biases and affect the overall integrity of the analysis. Therefore, addressing and managing these null values through appropriate techniques, such as imputation or data exclusion, is crucial to ensure the accuracy and reliability of our data exploration and subsequent analysis, To deal with this, we can use the following approach:

First, we cluster the jokes by vectorizing them using TF-IDF and applying dimensionality reduction (NMF).

Next, we select a particular user.

Now, there are two cases to consider:

Case 1 - If the user has rated all the jokes, we don't need to take any action as there are no null.

Case 2 - If there are some jokes that the user has not rated, we pick an unrated joke and determine in which cluster that particular joke is present. We then find the mean rating of that particular cluster as rated by the user, and assign that rating to this joke. We repeat this process for all unrated jokes.

Here is an intuitive explanation for this approach: When you cluster jokes and then provide ratings for null values based on the mean rating of jokes that a user has already rated within the same cluster, you are indeed preserving the user's preferences and maintaining consistency within the clusters.

This approach helps ensure that the imputed ratings align with the user's tastes and preferences to some extent.

After following the above approach we can reduce the count of null values from 320,000 to 120,000. Below is the plot for same

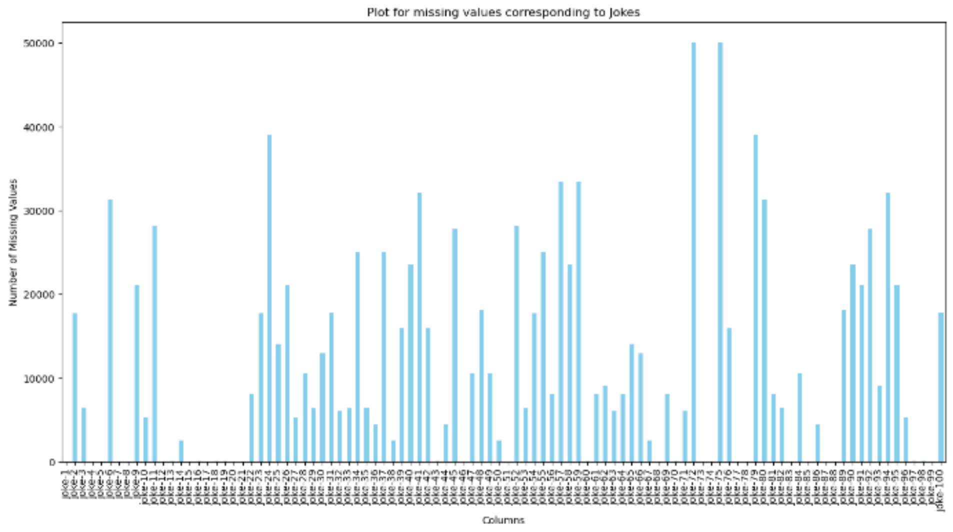

Figure 4: Plot for missing values corresponding to jokes

Then dealing with rest of Null values we use Imputer strategy, Impute by median.

Clustering and content-based similarity help in grouping similar jokes together. Imputing ratings based on the mean rating of the user within a specific cluster captures the thematic and content-related aspects of the jokes. By using this strategy, we aim to create a richer and more informative dataset that allows for a more comprehensive understanding of user interactions with the jokes. This approach ultimately enhances the quality and reliability of our data analysis and leads to more meaningful insights.

3.2 Data Visualization

Given that we have user ratings for each joke, we can categorize these ratings based on the mean rating into two distinct groups: 'good jokes' and 'bad jokes.

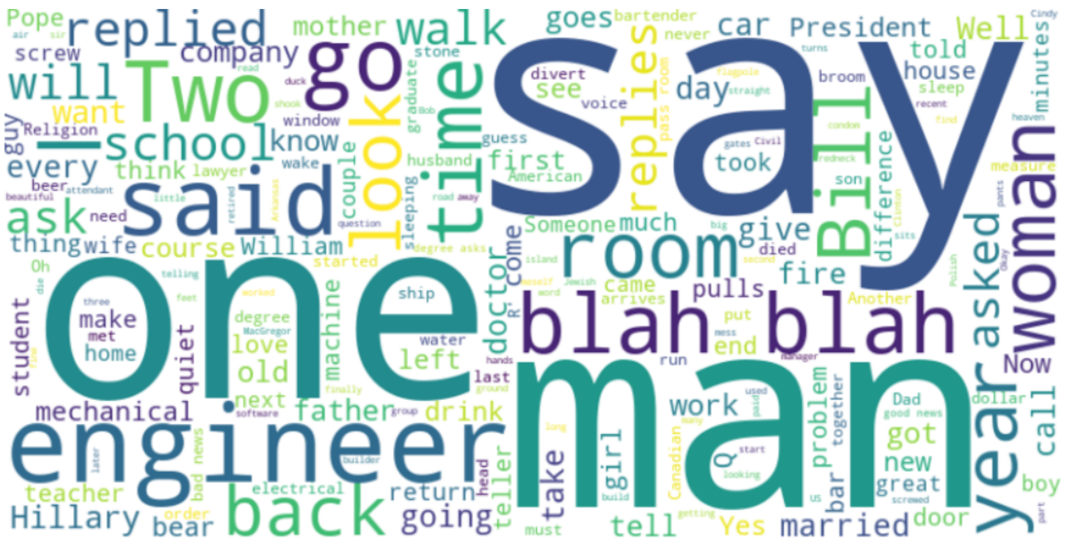

Figure 5: Good Jokes Word Cloud

The most prevalent words in the Good Joke Group include 'man,' 'one,' 'engineer,' 'said,' 'say,' and so on. The frequency of these words tends to increase with their size, making them more prominent within the Good Joke Group.

On the similar basis word cloud for bad jokes

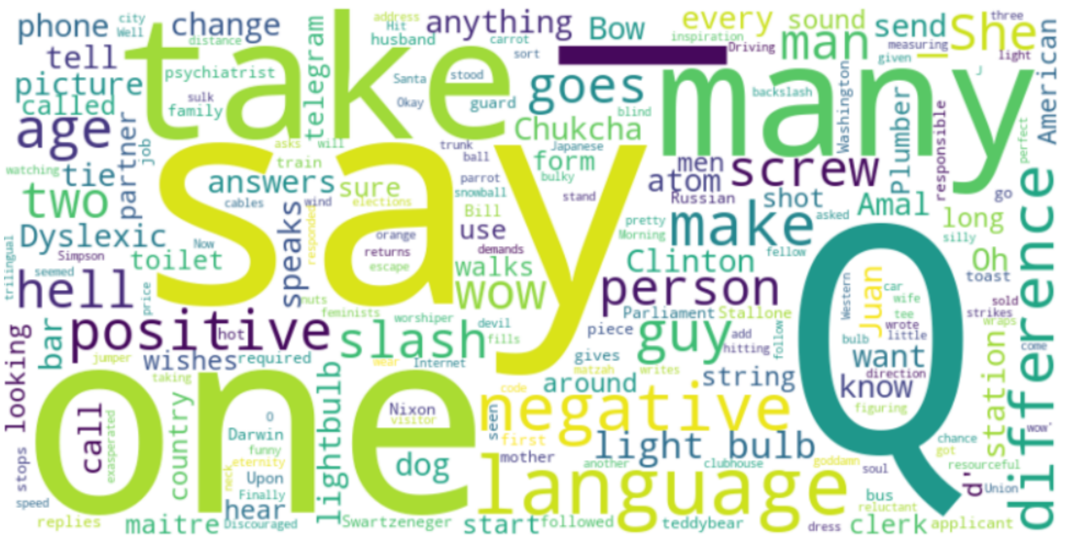

Figure 6: Bad Jokes Word Cloud

It's worth noting that in the Bad Joke Group containing jokes, a common theme emerges. Many of these jokes are question-answer-based, evident from the frequent use of 'Q' to start the jokes. Additionally, some common words such as 'positive,' 'hell,' 'one,' 'many,' and others appear regularly throughout the Bad Joke Group.

Furthermore, we extract specific features from the jokes and analyze how these features correlate with the average mean rating for each joke. This analysis allows us to uncover the relationships between joke characteristics and their ratings

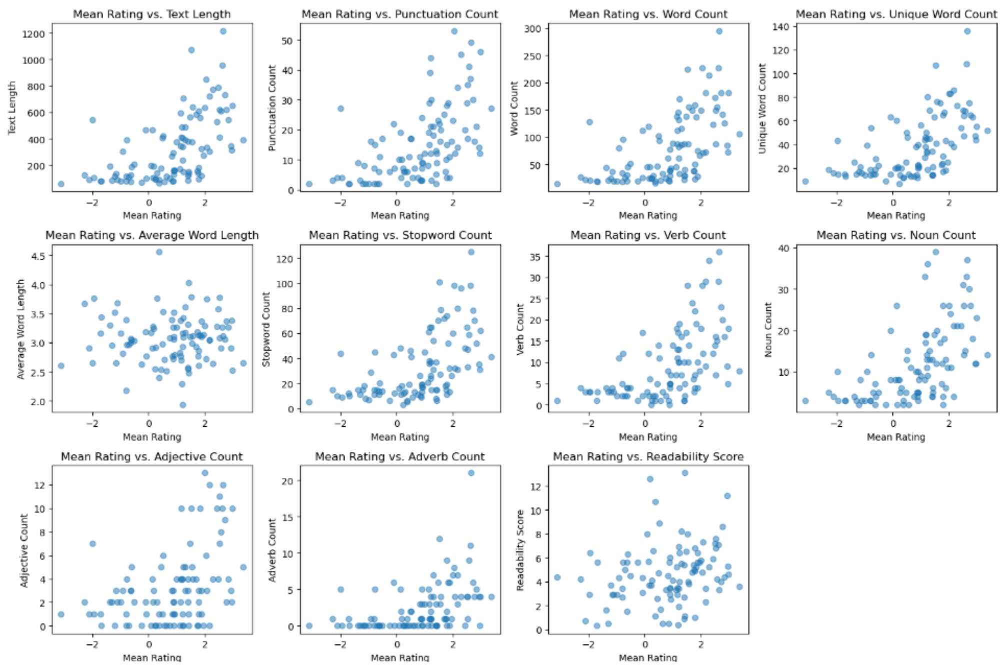

Figure 7: Features related to jokes

From the plot above, it's evident that nearly all features, except for 'Average Word Length,' 'Adverb Count,' and 'Readability Score,' exhibit a relatively linear relationship with the mean ratings of jokes.

For further analysis we can analyse statistical features. For user-item matrix

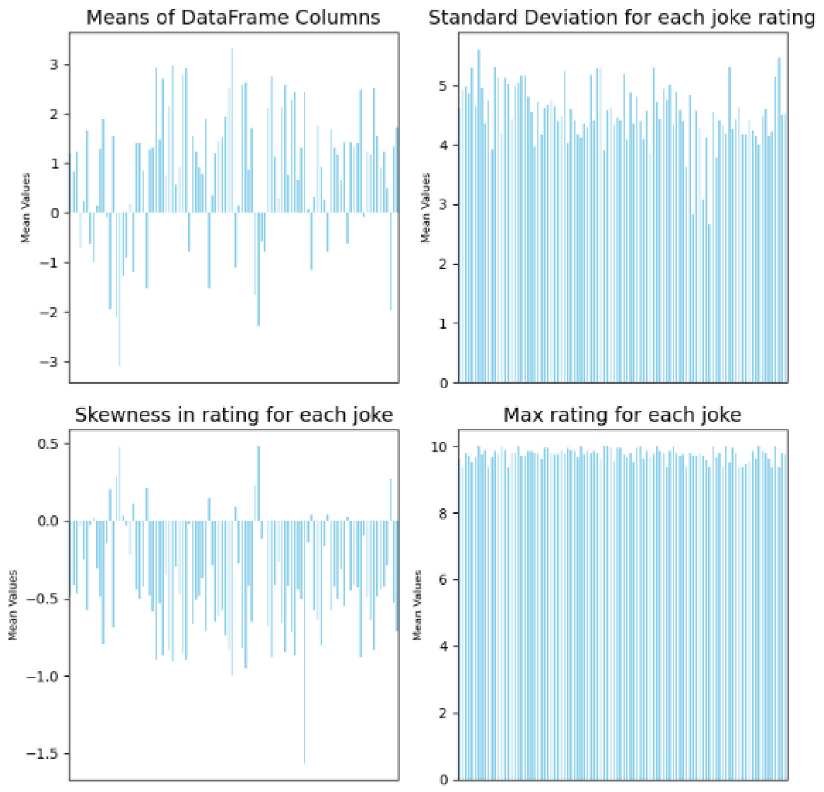

Figure 8: statistical features for user rating

Some Insight's from above that can help in building recommendation system:

i. There is no universal 'good' joke, as every user has their unique preferences, leading to individualized ratings for each joke. This diversity in user ratings is evident in the plot, where the maximum rating for nearly every joke reaches the upper bound, which is typically 10 in this recommendation system. This observation highlights the significance of every joke within the dataset, emphasizing that each one plays a crucial role in catering to specific user tastes and humor preferences.

ii. Understanding this skewness provides important context for our recommendation system. It suggests that users are more inclined to rate jokes they find amusing or favorable, and the dataset contains fewer ratings for jokes that users might not have enjoyed as much. This information can influence our recommendation algorithms and the way we handle the sparsity of the dataset, allowing us to make more informed decisions about how to tailor recommendations to individual users' preferences.

iii. In our dataset, we can identify a subset of 'bad' jokes, which consistently receive negative ratings on average. This finding suggests that there are common elements or patterns in these jokes that lead to unfavorable ratings. Analyzing these jokes can provide valuable insights into what makes a joke less appealing to users, shedding light on the characteristics or themes that are generally disliked.

Furthermore, the fact that these jokes have consistently negative average ratings highlights the need for caution when recommending similar jokes to users. Recommending jokes that share similarities with these universally 'bad' jokes may not align with user preferences and could lead to a poor user experience. Thus, understanding and excluding such jokes from the recommendation process is essential for delivering high-quality and enjoyable content to users.

Top 10 most rated jokes on an average rating:

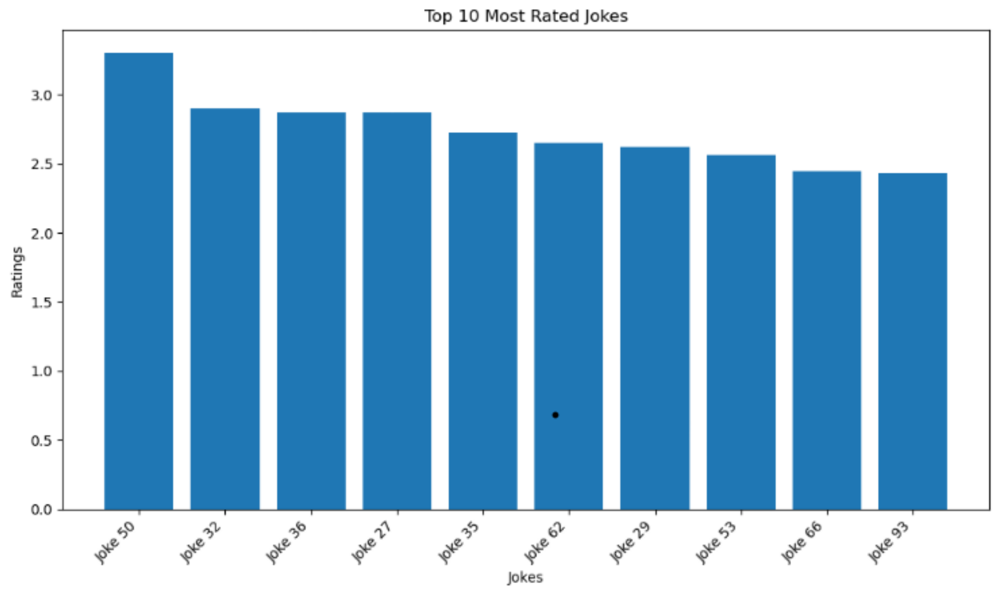

Figure 9: statistical features for user rating

Top 10 least rated jokes on an average rating:

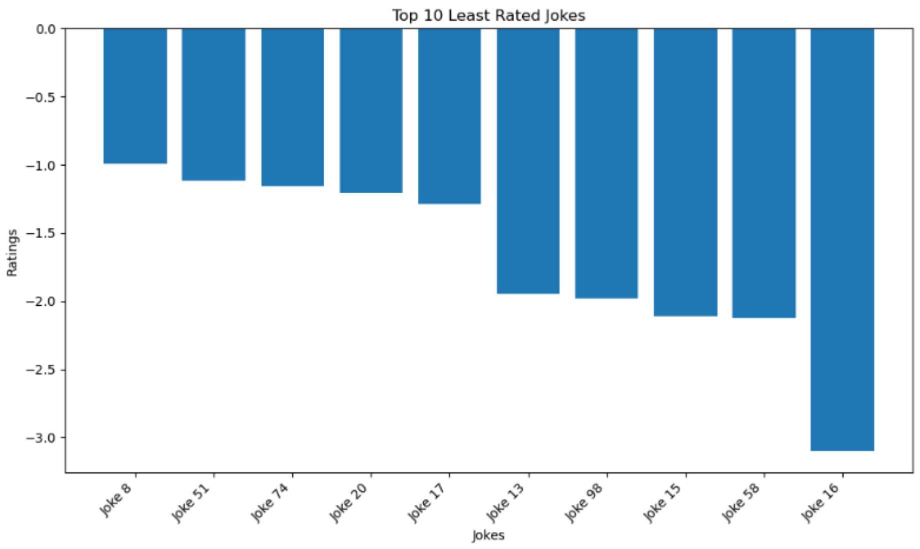

Figure 10: statistical features for user rating

3.3 Pattern Finding

The very first application of unsupervised learning that comes to one's mind is pattern recognition. Unsupervised learning algorithms play a crucial role in uncovering hidden structures or patterns within data. By identifying these patterns, unsupervised learning empowers us to gain a deeper understanding of the data and make data-driven decisions across various domains. we leverage unsupervised learning techniques in two distinct ways to gain insights from the given dataset:

i. Joke Clustering: We employ clustering algorithms to group similar jokes together. By doing so, we aim to uncover underlying themes, patterns, or topics within the jokes. This analysis helps us understand which jokes share common characteristics and are more likely to resonate with the same set of users.

ii. User Clustering: The second dimension of our unsupervised learning approach involves clustering users based on their ratings for the jokes. This user clustering provides a means to identify cohorts of users with similar humor preferences. By grouping users with comparable tastes, we can tailor recommendations more effectively, ensuring that users receive jokes that align with their unique sense of humo

Joke Clustering :

Converting the joke dataset from text format to numerical representations is essential since most machine learning algorithms require numerical inputs. To accomplish this conversion, we employ a structured pipeline that ensures seamless data transformation. This pipeline serves as a bridge between the raw text data and the numerical features required for effective machine learning. The conversion process is a crucial step that prepares the data for analysis and allows us to harness the power of machine learning algorithms for our recommendation system

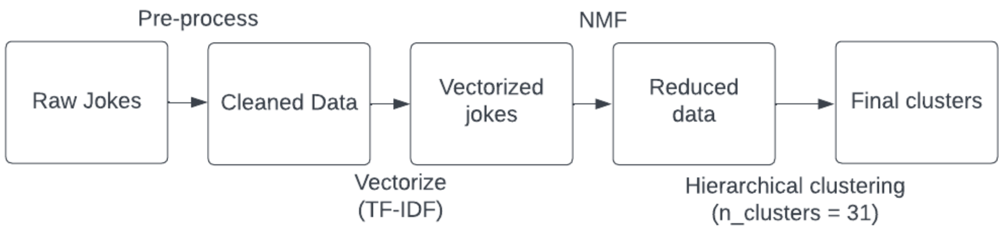

Figure 11: Block Diagram for clustering

After applying various clustering algorithms, we found that Hierarchical Clustering yielded the best results with an average silhouette score ranging from 0.55454 to 0.57534.

To determine the optimal number of clusters, we employed two common methods: the elbow method and the silhouette score plot. The elbow method, though attempted, did not yield a distinct ’elbow’ point, making it challenging to determine the ideal number of clusters. Consequently, we relied on the silhouette score plot to guide our cluster selection process. Here is the silhouette score vs n clusters plot

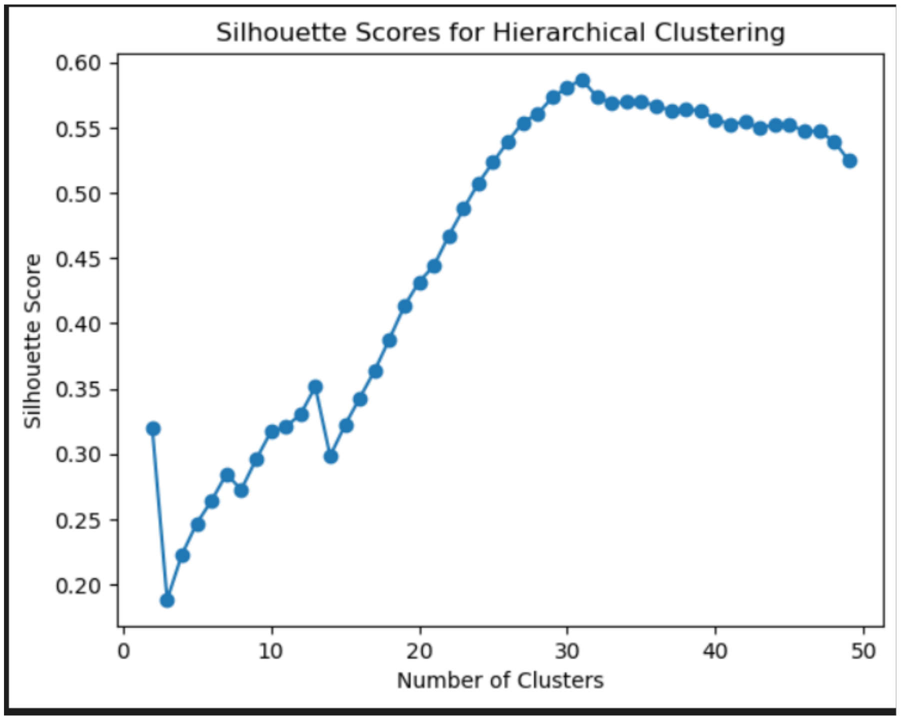

Figure 12: Silhouette Score plot vs n clusters

This is the final result for Hierarchical clustering:

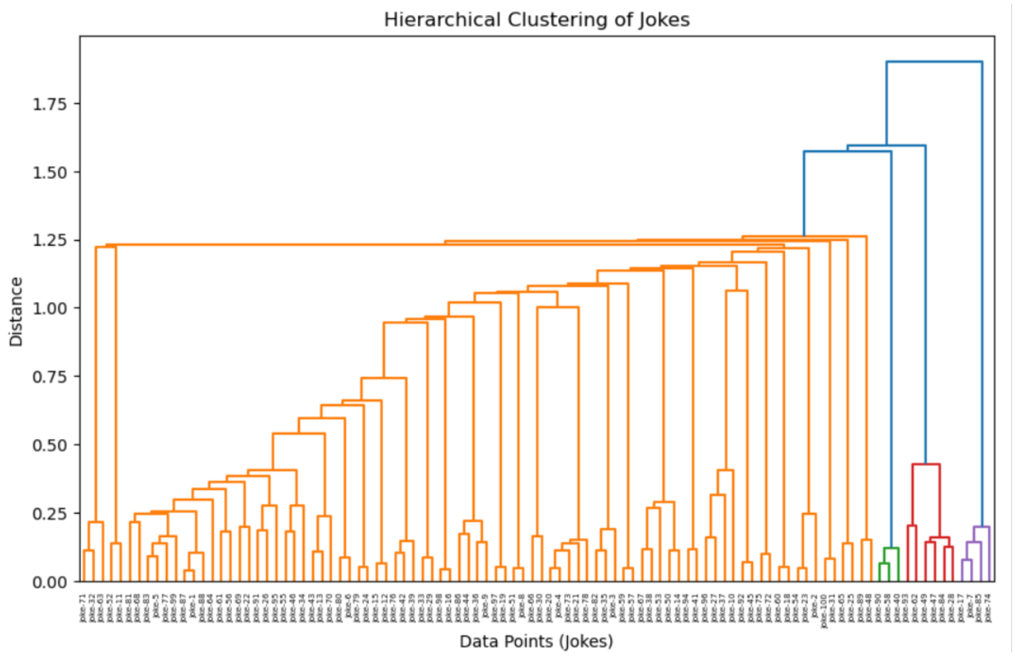

Figure 13: Hierarchical Clustering

In a similar vein, we applied the K-Means clustering algorithm with an optimal number of clusters set to 30 to 32, resulting in a silhouette score that closely matched the values mentioned in the code.

User Clustering :

For user clustering during the preprocessing stage, we implemented a comprehensive approach, which included the handling of null values to ensure data completeness. Given the dataset's substantial skewness, we carefully selected Singular Value Decomposition (SVD) as our dimensionality reduction method. SVD offers robustness and is well-suited to manage skewed data, making it a more reliable choice compared to Principal Component Analysis (PCA).

In the next step, we harnessed the power of K-Means clustering to group users with similar rating behaviors. To determine the optimal number of clusters, we employed the elbow method, which provides a data-driven approach for cluster selection. This meticulous process allowed us to create meaningful and user-centric clusters, facilitating the delivery of personalized recommendations that align with individual humor preferences.

Our user clustering methodology is not only a critical element of our recommendation system but also a testament to our commitment to data quality, robustness, and user satisfaction. By combining these techniques, we've laid the foundation for a powerful and user-focused recommendation engine.

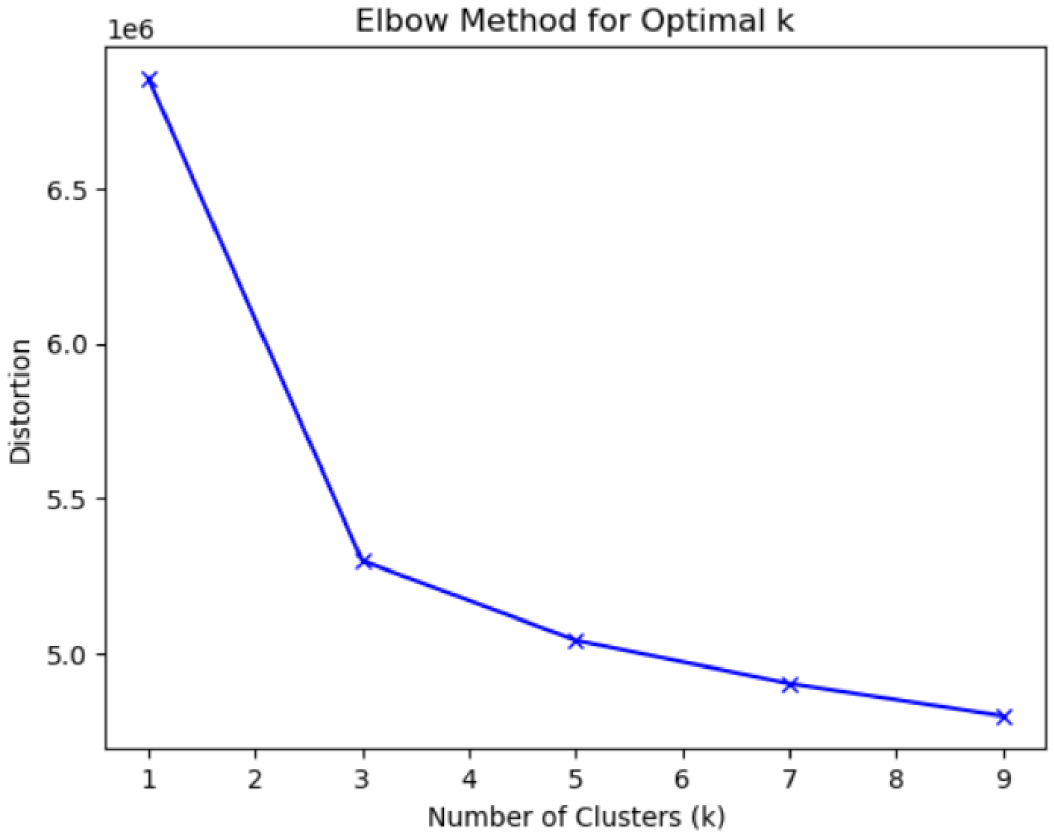

Figure 14: statistical features for user rating

Based upon this there are three cluster form, their is an clustering specific analyais based upon users rating

Based upon above clustering, top three most liked jokes in all three clusters are: joke-50

A guy goes into confession and says to the priest, "Father, I'm 80 years old, widower, with 11 grandchildren. Last night I met two beautiful flight attendants. They took me home and I made love to both of them. Twice."The priest said: "Well, my son, when was the last time you were in confession?"

"Never Father, I'm Jewish."

"So then, why are you telling me?"

"I'm telling everybody."

joke-27

Clinton returns from a vacation in Arkansas and walks down the steps of Air Force One with two pigs under his arms. At the bottom of the steps, he says to the honor guardsman, "These are genuine Arkansas Razor-Back Hogs. I got this one for Chelsea and this one for Hillary.

joke-53

One Sunday morning William burst into the living room and said, "Dad! Mom! I have some great news for you! I am getting married to the most beautiful girl in town. She lives a block away and her name is Susan."After dinner, William's dad took him aside. "Son, I have to talk with you. Your mother and I have been married 30 years.. She's a wonderful wife but she has never offered much excitement in the bedroom, so I used to fool around with women a lot. Susan is actually your half-sister, and I'm afraid you can't marry her."

William was heart-broken. After eight months he eventually started dating girls again. A year later he came home and very proudly announced, "Dianne said yes! We're getting married in June."

Again his father insisted on another private conversation and broke the sad news. "Dianne is your half-sister too, William. I'm awfully sorry about this."

William was furious! He finally decided to go to his mother with the news.

"Dad has done so much harm.. I guess I'm never going to get married," he complained. "Every time I fall in love, Dad tells me the girl is my half-sister."

His mother just shook her head. "Don't pay any attention to what he says, dear. He's not really your father."

The common joke in each cluster in: Joke-50:

A guy goes into confession and says to the priest, "Father, I'm 80 years old, widower, with 11 grandchildren. Last night I met two beautiful flight attendants. They took me home and I made love to both of them. Twice." The priest said: "Well, my son, when was the last time you were in confession?" "Never Father, I'm Jewish." "So then, why are you telling me?" "I'm telling everybody.

The least rated joke in each cluster are: joke-72: for cluster 1

On the first day of college, the Dean addressed the students, pointing out some of the rules: "The female dormitory will be out-of-bounds for all male students and the male dormitory to the female students. Anybody caught breaking this rule will be finded $20 the first time." He continued, "Anybody caught breaking this rule the second time will be fined $60. Being caught a third time will cost you a fine of $180. Are there any questions ?"At this point, a male student in the crowd inquired: *"How much for a season pass ?"*

joke-81: for cluster 2

An Asian man goes into a New York CityBank to exchange 10,000 yen for American Currency. The teller gives him $72.00. The next month the Asian man goes into the same bank with 10,000 yen and receives $62.00. He asks, "How come? Only $62.00?" The teller says "Fluctuations-Fluctuations!"Whereupon the Asian man looks back at the teller and says "*Fluk you Amelicans too!"*

joke-88: for cluster 3

A Czechoslovakian man felt his eyesight was growing steadily worse, and felt it was time to go see an optometrist.The doctor started with some simple testing, and showed him a standard eye chart with letters of diminishing size: CRKBNWXSKZY. . .

"Can you read this?" the doctor asked.

"Read it?" the Czech answered. *"Doc, I know him!"*

As we conclude this phase, we acknowledge that EDA is not merely about uncovering what the data holds, but also about asking the right questions, seeking context, and embarking on a continuous exploration of data dynamics. It's the compass guiding us through the labyrinth of information, helping us make informed choices, and inspiring the creation of a recommendation system that understands users, their preferences, and their humor.

4 Inference

The following are the inferences made from Existing Analysis and Exploratory Data Analysis.

The pre-processing techniques that can be used according to the existing analysis are either binary values or standardized data. If the data is stored with only binary values, it would make it easy to work with, but standardizing might improve the accuracy of the model by making the data diversified.

It can be further observed in our analysis that models which provide satisfactory results are collaborative filtering with Neighbours. For analyzing the performance of these models, Mean Absolute Error or sometimes Normalized MAE is used. For instance, the predictions made by Eigentaste have a normalized mean absolute error (NMAE) value of **0.187** and an accuracy as good as that of 80-NN, while being **100** times faster.

In this phase of constructing our recommendation system, our focus lies on data exploration and visualization to create a robust recommendation engine. We employ existing analysis, exploratory data analysis (EDA), and inference findings. While dealing with a dataset containing numerous missing values, we have detailed our strategies for imputation.

Our approach involves the identification of data patterns using unsupervised learning methods like K-Means and Hierarchical Clustering. For dimensionality reduction, we have utilized Non-Negative Matrix Factorization (NMF) and Singular Value Decomposition (SVD) since PCA is not suitable for this dataset. EDA has unveiled various patterns, and our inference findings highlight key features for model building.

Furthermore, meaningful insights have emerged from our analysis. It's evident that all jokes are liked by some users, but there are jokes with, on average, negative ratings. This implies that these jokes may contain sensitive content. Any new joke falling within these clusters should be handled with caution. Additionally, patterns have been observed through the mean rating versus statistical features of a joke (such as its length, number of adverbs, pronouns, and adjectives). They exhibit a linear relationship that can be leveraged during modelling.

Lastly, skewness provides a crucial context for our recommendation system. It indicates that users are more inclined to rate jokes they find amusing or favourable. The dataset contains fewer ratings for jokes that users might not have enjoyed as much. This information can significantly influence our recommendation algorithms and our approach to dealing with the dataset's sparsity. It enables us to make more informed decisions on tailoring recommendations to individual users' preferences.

5 Methodology

There are two types of methodologies considered, one based on **User Based Rating** and another one based on **Average Ratings** of jokes were considered.

In User Based rating, we predict the rating given by every user for a new joke and in Average Ratings approach, we get the average rating of a new joke which determines how liked a joke is on average among all given users. Both these methods vary in Pre processing and Feature extraction. The methodologies taken for both these cases are seperated in the following text.

5.1 User-Rating Data Pre-Processing

5.1.1 *Average Joke Ratings model*

The Jester Dataset-1 had three different user rating data sets are combined into a single data frame. All the 99.0 ratings denote jokes that have not been recommended by the user and are replaced with NaN values.

5.1.2 *User-Based Joke Ratings model*

Similiar to the Average joke ratings model, the three different user rating data sets are combined into a single data frame. All the 99.0 ratings denote jokes that have not been recommended by the user and are replaced with NaN values. Singular value decomposition is applied to factorize the user-item matrix obtained above. The predicted matrix is taken as the dot product of the matrices obtained during SVD. The matrix is clipped between -10 and 10. To check the accuracy, MSE is calculated to measure the difference between predicted and original matrices. The MSE value was found to be **15.57**

5.2 Feature Engineering

5.2.1 *Average Joke Ratings model*

The jokes are extracted from the HTML files and cleaned by removing pipe characters and hyphens. They are then split into training and testing data with a 90:10 ratio. Jokes are then striped of any special and whitespace characters. They are tokenized and stop words are removed.

Two text normalization techniques, stemming and lemmatization to reduce words to their base or root words have been tried. Stemmming reduces the word by removing the last few characters, which can often lead to misspelled and incorrect root words. Lemmatization reduces words to its root meaning based on its meaning (Lemma).

Here, lemmatization was preferred and the cleaned jokes were processed accordingly. Further, the TF-IDF values for these pre-processed jokes were calculated.

Part-Of-Speech Tagging was also done and applied to the preprocessed jokes to segregate words into grammatical categories that can be used to identify their purpose in each sentence.

After this, the features were extracted for model training, with the following features:

- **Irony detection** is done by checking if the joke has a negative sentiment polarity. If it is, then it is marked as irony, otherwise it is not.
- **Sentiment analysizer** evaluates the emotion associated with the joke and returns scores indicating whether it is a positive, negative or neutral sentiment.
- **Structural features** like average sentence length, maximum sentence length, punctuation count and the number of paragraphs in a joke.
- **Humour features** like wordplay and incongruency are detected using the POS tags and looking at the lemma and dependency tags for the tokens in the joke.
- **POS distributions** is calculated by counting the number of tokens associated with that POS.
- **Lexical features** is calculated by taking the ratio of unique words to number of words in the joke, giving vocabulary richness and the ratio of unique words to number of words, including numbers.
- **Stylistic features** analyzes text for readability metrics and gives insights into text complexity and informal (jargon or slang words) language use.

5.2.2 *User-Based Joke Ratings model*

Phrases and Phraser models are created using Gensim to identify bigrams and trigrams. The Dictionary and Corpus are prepared for LDA (Latent Dirichlet Allocation) modeling, and the LDA model is trained on the corpus to extract topics. NMF (Non-Negative Matrix Factorization [8]) was also tried to reduce the dimension of the texts, but it reported a lower R squared score, so it was dropped.

Text analysis is performed as a part of Sentiment Analysis. Features extracted by this process are text length, punctuation count, word count, unique words, verb count, noun count and sentiment score. This process is repeated for every joke.

5.3 Model

5.3.1 *Average Joke Ratings model*

A few different models were tested to obtain the best possible R-sqaured value.

- **Ada Boost:** It uses a base weak learner and tries to boost the performance of a weak learner by iteratively shifting focus towards flawed observations that were difficult to predict. Finally, a strong learner is formed by a weighted addition of weak learners. At each step, weight of the wrongly classified data points is increased and the weights of correctly classified data points is decreased.
- **SVR with different parameters (c=1 and c=2):** SVM's regression model is referred to as Support Vector Regression. In this, the best fit line is the hyperplane which contains the maximum number of points. It allows flexibility to choose an acceptable error margin during regression. The model is tested with different values of the regularization parameter, the degree of importance assigned to misclassifications.
- **Ridge regressor and Bayesian ridge regressor:** Ridge Regressor is a linear regression technique that prevents overfitting by adding a regularization term, adding stability. The regularization term is fixed. Bayesian Ridge Regressor incorporates Bayesian principles, providing a probabilistic view of the regression coefficients and their uncertainties. The regularization term is obtained by a distribution estimated from the data, which allows for a nuanced understanding of uncertainty.

R-square scores were calculated for each of them, using K-Folds validation.

5.3.2 *User-Based Joke Ratings model*

There is no particular model training done for user-based method. In order to test the jokes, the test set jokes are first transformed, and its representation is compared to representation of jokes in training set using cosine similarity. It is a metric that measures the cosine of the angle between two vectors. In the context of recommendation systems, it's used to quantify the similarity between jokes. A higher cosine similarity indicates greater similarity between vectors.

5.4 Testing

5.4.1 *Average Joke Ratings model*

The model's fit was measured using K-Fold cross-validation with K=3 and average R-squared score. K-Fold with K=3 divides the data in three equally sized folds. The model is trained three times, each time using two folds as the training set and one fold as the testing set. R-squared score is calculated for each fold and the average performance is calculated.

5.4.2 *User-Based Joke Ratings model*

During testing, the similarity between target jokes and others is calculated using a similarity measure like cosine similarity. Most similar jokes are selected based on content features.

The cosine similarities act as weights. Weighted average ratings are calculated for each user in the training set based on their ratings for the most similar jokes. This is to give more weight to the ratings from users whose preferences are more similar to the current test joke. These ratings provide a prediction for the target joke's rating, determined by the average or weighted average rating of the selected similar jokes.

6 Results

Upon analysis, it was found that average joke ratings model performed much better than the user based joke ratings model. It was decided to move forward with this model, not only because the R-square scores are higher but also since that it aligns better with the problem statement. The results obtained by both have been documented below.

6.1 Average Joke Ratings model

The R-squared values for the different models, using K-Folds for validation, are given as follows:

Model	Fold 1	Fold 2	Fold 3	Average R2 score
Ada Boost	0.172	0.255	0.471	0.299
Ridge	0.285	0.254	0.280	0.273
Bayesian Ridge	0.341	0.282	0.430	0.351
SVR, c=1	0.352	0.248	0.437	0.346
SVR, c=2	0.359	0.285	0.418	0.354

Table 1: R-squared scores for different models

Hence, the highest R-square value was obtained by SVR with c=2 with an average R-sqaured value of **0.354**. R-sqaured is a measure of goodness of fit. A higher value of R-squared indicates that the model fits the data well. Here, an R-sqaured score of **0.354** suggests that it on average predicts 35 percent of the relationship between independent and dependent variables.

6.2 User-Based Joke Ratings model

The User-Based model gave relatively poorer performance during testing as the user rating dataset obtained is sparse with multiple **NaN** values which affects our prediction. The R-squared value obtained when LDA was used was around **0.053** and for NMF it was **0.034**.

7 Comparison with Existing Analysis

In the existing analysis, the techniques recommended for tackling joke recommendation system are only valid for suggesting an unseen joke to a user, from the current set of jokes already rated by other users. To handle the problem of recommending new jokes, it is must that the new joke is either already rated by some other user or it should have some correlation with the jokes already present.

These assumptions are not guaranteed and hence it was decided to opt for a Content Filtering focused approach that extracts information and features from the Joke text. Hence the results vary from the existing analysis.

In the user based rating approach, cosine similarities was used to make prediction. This resulted in a weak accuracy, and significantly lower than the existing analysis. Upon taking average rating method, the accuracy was significantly improved, comparing closely to the values obtained in the existing analysis.

8 Model Complexity

Complexity is a pivotal consideration in machine learning model development. Striking the right balance is crucial—too simple may overlook nuances, while excessive complexity risks overfitting. It involves thoughtful choices in model architecture, parameters, and algorithms. Effectively managing complexity is an art that requires a deep understanding of the data and problem at hand, aiming for a delicate balance that maximizes performance and generalizability . In addition to model intricacies, the complexity of a machine learning system extends to various stages of development, encompassing data preprocessing, feature engineering, and the selection of evaluation metrics. These decisions collectively shape the overall complexity and effectiveness of the system. Practitioners must navigate this complexity with a keen awareness of computational constraints, interpretability requirements, and the need for the model to adapt to unseen data. Thus, achieving an optimal level of complexity involves a careful, iterative process that combines domain expertise with a nuanced understanding of the algorithmic landscape, ultimately leading to models that not only perform well but are also robust and scalable in real-world applications.

ML model complexity can be analyzed during training time and testing time

a. Training Time: During training, we preprocess jokes, apply lemmatization, extract features from them, and then feed these features to the SVR (Singular Value Regression) for training. To calculate the overall complexity, we need to analyze the complexity at each of these steps. So let's assume we have M jokes, and each joke has an average length of N. Let's break down our preprocessing. In this, we are performing three steps:

1. Remove tags and punctuation.
2. Remove stop words and tokenized.
3. Convert the string to lowercase.

For each joke, all these steps are performed, and the worst-case complexity for each joke to go through each step is O(N). Since this process is repeated for M jokes, the overall worst-case complexity for this step is O(M * N). The next step that we implement is lemmatizing each joke. This step also takes O(M * N) because for a single joke, it takes O(N) worst time to lemmatize it. The next step is Feature Engineering, where for each joke, we extract nearly 25 features, and they are calculated. Again, we are doing the same thing, looping over jokes and extracting all the features. Extracting each feature takes O(N). Having 25 features and looping over M jokes, the overall time complexity for this is O(25 * M * N) = O(M * N).

Up to this point, our final dataframe is ready, which has M jokes and 25 features. In the model training part, we apply SVR to our final dataset, which takes O($M\hat{2}$ * 25) = O($M\hat{2}$).

Hence, the final complexity for training is: O(M * N + M * N + M * N + $M\hat{2}$) = O(M * (N + M)).

The final training complexity is O(M * (N + M)).

b. Testing Time: Similar to training, here also jokes are preprocessed, lemmatized, and feature extraction is done. Based on the above analysis, its time complexity is O(M * N). Now, during prediction time, each joke SVR takes O(M * 25) = O(M).

So the final time complexity for prediction is: O(M * N + M * N + M * N + M) = O(M * N).

9 Conclusion

In conclusion, this report endeavors to construct a recommendation system designed to suggest ratings for newly introduced jokes. The foundation of our recommendation system lies in the application of Natural Language Processing (NLP) and its associated concepts, including sentiment analysis, topic modeling, Word to Vector utilizing TF-IDF, and Bag of Words. Through the extraction of various features from raw texts, we aim to enhance our model's ability to generalize effectively across a spectrum of jokes.

Subsequently, we systematically apply diverse algorithms, evaluating their performance to make an informed selection. Based on our findings, Support Vector Regression (SVR) with a parameter setting of C = 2 emerges as the most suitable choice. Finally, we conduct a thorough analysis of model complexity, refining our understanding of the system's intricacies and optimizing its performance. This comprehensive approach ensures the robustness and efficacy of our recommendation system for rating newly introduced jokes.

References

[1] Navoneel Chakrabarty. Cluster analysis on jester dataset: A review. *arXiv preprint arXiv:2110.02740*, 2021.

[2] Navoneel Chakrabarty, Srinibas Rana, Siddhartha Chowdhury, and Ronit Maitra. *RBM Based Joke Recommendation System and Joke Reader Segmentation*, pages 229–239. 11 2019.

[3] Joana Costa, Catarina Silva, Mário Antunes, and Bernardete Ribeiro. Get your jokes right: Ask the crowd. In Ladjel Bellatreche and Filipe Mota Pinto, editors, *Model and Data Engineering*, pages 178–185, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[4] Kenneth Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4:133–151, 07 2001.

[5] Soumya Agrawal. JOKE RECOMMENDER SYSTEM USING HUMOR THEORY. 7 2020.

[6] Wen Wu, Liang He, and Jing Yang. Evaluating recommender systems. In *Seventh International Conference on Digital Information Management (ICDIM 2012)*, pages 56–61. IEEE, 2012.

[7] Abhijeet Anand. Jester-Joke-Recommender-System, 2018.

[8] Rob Salgado. Topic modelling, 2020.