Jester-Joke Recommendation System

Harshit Sharma 2021463 | **Hardik Singh** 2021390 | **Shreyas Gupta** MT23221 | **N Deepika** MT23048

Abstract

In this machine learning project, we explore the realm of humor by leveraging the Jester Joke Dataset, a collection of user-rated jokes. The primary objective is to design and implement a model using classical machine learning techniques to predict user ratings for jokes. The dataset consists of 100 jokes, each rated by a diverse group of users. Our approach involves training the model on this comprehensive dataset, enabling it to learn patterns and relationships between joke content and user preferences.

The methodology encompasses various steps, starting with extensive data preprocessing, including text cleaning, feature extraction, and the creation of N-gram representations to capture the nuances of joke content. Classical machine learning algorithms, such as Support Vector Machines (SVM) and Linear Regression, are employed to build a predictive model. The model is fine-tuned and validated to ensure robust performance across a range of jokes.

To evaluate the model's effectiveness, a test dataset comprising 20 jokes is reserved for prediction. Users are tasked with rating these jokes based on their personal preferences, and the model's predictions are compared against the actual user ratings. Performance metrics such as Root Mean Squared Error (RMSE) is utilized to assess the accuracy and reliability of the model.

This project not only explores the application of classical machine learning techniques in humor modeling but also introduces a practical scenario where the trained model predicts user ratings for a novel set of jokes. The results and insights gained from this endeavor contribute to our understanding of computational humor modeling and its potential applications in entertainment and recommendation systems.

1 Introduction

Humor, a multifaceted and subjective aspect of human interaction, has intrigued scholars and enthusiasts for centuries. In the digital age, the exploration of humor extends into the realm of machine learning, where algorithms attempt to decipher and predict human preferences for jokes. This project delves into the world of computational humor modeling, utilizing the Jester Joke Dataset—a comprehensive collection of user-rated jokes. The primary goal is to construct a robust model employing classical machine learning techniques that can effectively predict user ratings for jokes.

The dataset, comprising 100 jokes, provides a diverse array of humor styles, themes, and tones, each evaluated by a multitude of users. Leveraging classical machine learning algorithms such as Support Vector Machines (SVM) and Linear Regression, our project seeks to unravel the intricate relationships between joke content and user preferences. Through extensive data preprocessing, including text cleaning and the creation of N-gram features, we aim to distill the essence of humor into quantitative representations.

The approach extends beyond the training phase, incorporating a real-world scenario where the trained model faces a set of previously unseen jokes. This test dataset, consisting of 20 jokes, serves

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

as a litmus test for the model's predictive capabilities. Users are invited to rate these jokes based on personal preferences, allowing us to assess the model's ability to discern humor patterns and accurately predict user sentiments.

This project not only contributes to the field of computational humor but also sheds light on the applicability of classical machine learning techniques in modeling subjective preferences. As humor remains an integral aspect of social interaction, understanding its computational facets has implications for entertainment recommendation systems, sentiment analysis, and beyond. The subsequent sections will detail the methodology, experiments, and findings, offering insights into the fascinating interplay between humor, machine learning, and user preferences.

2 Literature Review

2.1 CF Algorithm

The "Collaborative Filtering" uses the preferences of a set of users to predict the preferences of a new user, i.e., for new users, recommendations are based on their preference predictions. In users' collaborator systems, each rating improves the performance of the overall system. The fundamental assumption is that if users A and B rate k items similarly, they share similar tastes and hence will rate other items similarly.

A new CF algorithm, Eigentaste, applies PCA to a dense subset of the rating matrix. Eigentaste also uses universal queries to elicit real-valued user ratings on a common set of items, where PCA makes dimensionality reduction in offline clustering for users and rapid online cluster assignment.

2.2 POP Algorithm

In this recommendation, we treat all users coming from the same pool and then base the recommendations based on the global mean rating. Here, the training set is used to find the global average, while the test set is used to evaluate the predictions. POP will predict the rating for every joke based on the global mean value and then do a recommendation

2.3 Comparision Metric : NMAE

This metric if widely used in CF literature, where if pij is prediction for user i on rating an item j, then MAE for user i is :

$$MAE = \frac{1}{c}\sum_{j=1}^{c} |r_{i_j} - p_{i_j}|$$

Here c is the number of items rated by user i. We can normalize this error for a given range to get NMAE ie Normalized Mean Absolute Error

$$NMAE = \frac{MAE}{r_{max} - r_{min}}$$

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

2.4 Existing Work

We can classify CF algorithms into two classes: Memory-based and Model- based. Memory-based algorithms operate over the whole user database to make predictions, like the commonly used Nearest-neighbors which use different distance measures to relate users.In Model-based the systems learn from a compact model inferred from the data. The Eigentaste can be considered Model-based.

In Personality Diagnosis (PD), latent variable computes the probability of a new user is for an underlying "personality type," and then these user preferences are a manifestation of this personality type. For a personality type of a user, PD finds the probability that the given user has same personality type as other users in the system, and so probably that user will like some new related item.

An axiomatic based CF approach can also be followed.

An agent-based approach to CF, will use several algorithms that combine ratings data with other sources of information such as the geographic location of the user. Then Weighted majority voting is used to combine recommendations from different algorithms

In neighbor-based CF algorithms there are three steps:

1. weighting of possible neighbors,
2. then selecting neighborhoods and,
3. producing a prediction from a weighted combination of neighbors ratings.

This can use Spearman (rank-based) correlation weighting as an alternative to Pearson correlations and a "significance weighting" based on the number of items two users have rated in common. The Receiver Operator Characteristic (ROC) sensitivity can be used to measure accuracy.

In CF methods we have the problem of sparseness i.e. many values in the ratings matrix are null since all users do not rate all items, so calculating the distances between users becomes complicated since the number of items users have rated in common is not constant. One can address this by inserting global means for null values or use Singular Value Decomposition for case of significance weighting. SVD reduces the dimensionality of the ratings matrix and identifies latent factors in the data

Eigentaste addresses the sparseness problem with universal queries(presents each user with the same gauge set of items to rate initially. Each query contains a quick unbiased summary so that users can form an immediate opinion) instead of user-selected queries. Universal ratings allows the system to collect immediate feedback on all recommended items. Eigentaste trains on user taste with universal query. The resulting matrix from gause set is dense in nature so, we calculate the square symmetric correlation matrix and then apply PCA, as it reduces dimensionality by optimally projecting highly correlated data along a smaller number of orthogonal dimensions

3 Data Description

The project incorporates two primary datasets: the Jester dataset and a complementary joke-rating dataset, collectively contributing to a comprehensive study of humor-related content.

3.1 Jester Dataset:

Data Source: The Jester dataset is a collection of humor-related textual content stored as HTML files. It includes 100 HTML files, each corresponding to a different class of humor content. These files house jokes of varying lengths extracted from diverse sources.

Data Format: The jokes are encapsulated as plain text within HTML files, often containing HTML tags. Preprocessing is necessary to remove these tags for effective analysis.

Data Variability: The dataset exhibits diversity in joke length, style, and humor categories, providing an extensive palette for exploring various facets of humor and natural language processing.

Data Quantity: In total, the Jester dataset encompasses 100 jokes, individually housed in separate HTML files.

Data Quality: Cleaning and preprocessing efforts are essential to eliminate HTML tags and irrelevant information, ensuring the dataset's readiness for machine learning tasks. As part of this process, we extracted jokes from HTML sources and organized them into a CSV file.

Data Potential: The Jester dataset holds considerable potential for an array of humor-related machine learning applications, including humor classification, sentiment analysis, and humor recommendation systems.

`https://eigentaste.berkeley.edu/dataset/jester_dataset_1_joke_texts.zip`

3.2 Joke-Rating Dataset:

Data Source: The joke-rating dataset complements the Jester dataset by providing a quantitative perspective on the humor content. It contains ratings, typically on a scale, that reflect the perceived humor of the jokes.

Data Format: This dataset usually consists of structured tabular data, with columns representing joke IDs, user IDs, and corresponding joke ratings.

Data Variety: The joke-rating dataset is a valuable resource for examining user-generated humor preferences, enabling the development of recommendation systems and understanding the subjective nature of humor.

Data Quantity: The exact size of the joke-rating dataset can vary, but it typically comprises a substantial number of ratings associated with the jokes in the Jester dataset.

Data Quality: Preprocessing may involve handling missing or inconsistent ratings, aligning them with the jokes in the Jester dataset, and addressing potential data quality issues.

Data Potential: The joke-rating dataset offers the means to build humor recommendation models, gauge humor perception, and explore patterns in user-generated humor preferences.

`https://eigentaste.berkeley.edu/dataset/jester_dataset_1_1.zip`

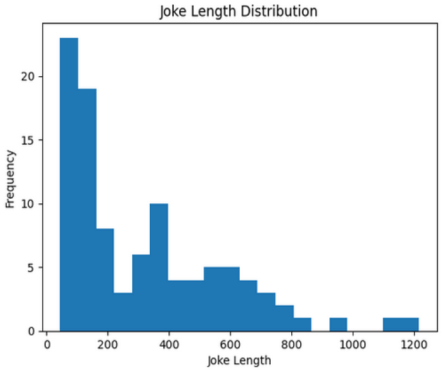
`https://eigentaste.berkeley.edu/dataset/jester_dataset_1_2.zip`

`https://eigentaste.berkeley.edu/dataset/jester_dataset_1_3.zip`

This combined dataset description highlights the distinctive characteristics and potential applications of both the Jester dataset and the joke-rating dataset, providing a robust foundation for subsequent analyses and machine learning tasks.

4 Exploratory Data Analysis

4.1 Joke Length:



count	100.00
mean	320.64
std	255.43
min	45.00
25%	109.25
50%	225
75%	471.25
max	1216.00

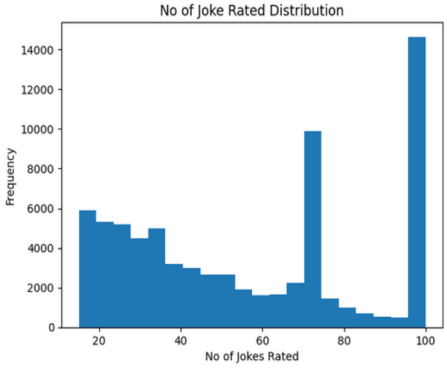
Fig 5. Distribution of Ratings per Joke

Upon examining the joke lengths in the dataset, the following insights emerged:

1. Majority of Short Jokes: Half of the jokes are 225 characters or less, and 75% fall within the range of 45 to 471.25 characters, while 25% of the jokes lie between 471.25 and 1216, indicating that the majority of jokes in the dataset are relatively short.
2. Positive Skewness: The skewness value of 1.177 suggests a positively skewed distribution. This means there are more short jokes in the dataset than long ones.
3. Variability: The dataset exhibits a notable standard deviation of 255.43, indicating considerable variability in joke lengths. This diversity allows for a range of joke lengths to cater to different user preferences.

In summary, the data reflects a prevalence of short jokes but also offers a wide variety of joke lengths, accommodating diverse user preferences in the recommendation system.

4.2 Number of Jokes rated by users:



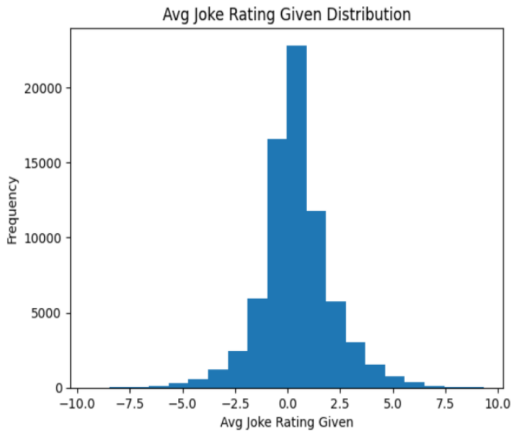
count	73421.00
mean	56.34
std	29.02
min	15.00
25%	29.00
50%	52.00
75%	75.00
max	100.00

Fig 2. Distribution of Number of Jokes Rated by a User For the 100 Jokes

The analysis of the number of jokes rated by users reveals two significant phenomena:

1. Bimodal Engagement: The data displays a bimodal distribution, with two prominent modes. The first mode occurs at the maximum rating of 100 jokes, indicating a substantial group of highly engaged users. The second mode is observed around 71 to 74 ratings, representing another significant cluster of moderately engaged users.
2. Diverse Engagement Levels: In addition to the modes, there is a wide range of user engagement levels. Users are found both at the extremes (rating all jokes) and in the middle range, from 15 to 100 jokes rated.

4.3 Average Joke Rating Given by Users:



count	73421.00
mean	0.42
std	1.70
min	-9.39
25%	-0.43
50%	0.31
75%	1.21
max	9.31

Fig 3. Distribution of Average Rating Given by a User

The analysis of average joke ratings provided by users reveals:

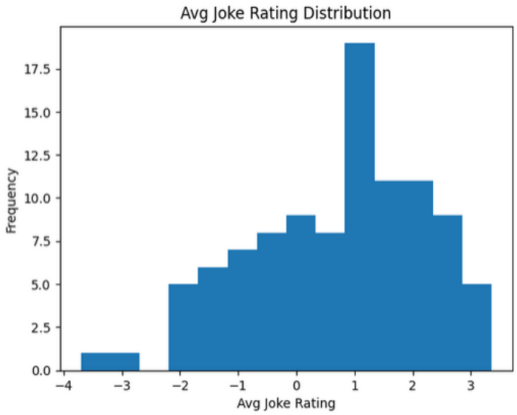
1. Slight Positive Sentiment: Users, on average, express a slight positive sentiment towards jokes, with a mean rating of approximately 0.418.

2. Variability in Ratings: However, the dataset exhibits a wide range of ratings, encompassing both highly positive and negative assessments, reflecting diverse user preferences.

3. Balanced Distribution: The distribution of ratings is relatively balanced, centered around the mean rating, with a slight positive skew.

In summary, users generally rate jokes positively, but there is considerable variability, reflecting diverse user opinions about humor. The distribution maintains a sense of balance, highlighting the need for a recommendation system that caters to a broad spectrum of user preferences.

4.4 Average Joke Rating:



count	100.00
mean	0.70
std	1.49
min	-3.70
25%	-0.42
50%	1.00
75%	1.79
max	3.36

Fig 3. Distribution of Average Rating Given by a User

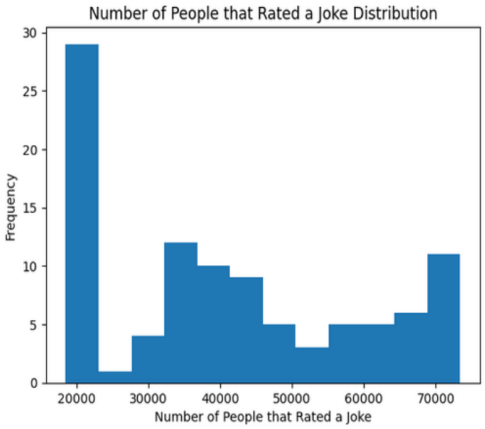
The analysis of average ratings received by jokes reveals:

1. Slightly Positive Reception: Jokes, on average, receive slightly positive ratings with an average of 0.7, suggesting a generally favorable reception by users.

2. Varied Range of Ratings: The average ratings span from -3.7 to +3.36 within the range of +10 to -10. This diversity indicates that jokes elicit mixed responses, with none receiving extremely negative or positive ratings. It underscores the subjectivity and variability in humor perception.

In summary, jokes tend to be positively received on average, and the diverse range of ratings showcases the nuanced and varied responses of users to different jokes. These insights are valuable for developing a recommendation system that caters to a wide spectrum of humor preferences.

4.5 Number of People that Rated a particular joke: :



count	100.00
mean	41363.60
std	18341.64
min	18505.00
25%	21941.25
50%	39103.50
75%	55617.50
max	73413.00

Fig 3. Distribution of Average Rating Given by a User

The analysis of user ratings for each joke reveals:

1. Widespread User Participation: Jokes, on average, attract a substantial number of ratings, indicating active user engagement in the rating process.
2. Varied Popularity: There is considerable variability in the number of people rating each joke, reflecting differences in the jokes' popularity among users.
3. Concentration of Ratings: The 25th percentile value of 21,941 reveals that 25% of jokes are rated by a relatively smaller range of 18,505 to 21,941 people. This suggests that a substantial portion of jokes garners ratings from around 20,000 individuals.
4. Sharp Decline in Popularity: An intriguing observation is the sharp decline in the number of people rating jokes from the 71st joke onwards, continuing until the 100th joke. This decline could indicate that some users may not have been exposed to jokes beyond the 70th joke, potentially due to dataset limitations or user preferences.

In summary, the analysis highlights user engagement in rating jokes, diverse joke popularity, and a potential limitation in user exposure to jokes beyond a certain point. These insights inform the design of a recommendation system that accommodates varying levels of joke popularity among users.

4.6 Correlation Heatmap:

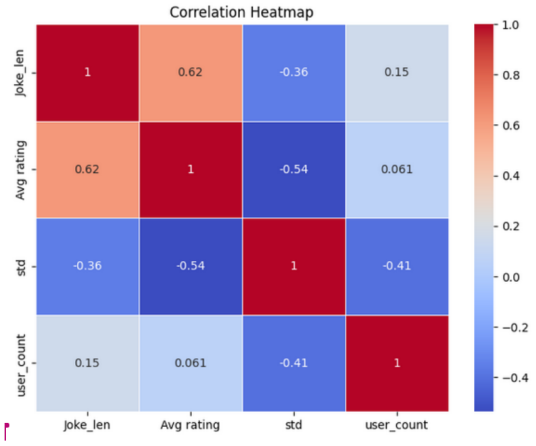


Fig 7. Correlation Heatmap of Dataset Variables

Some Inferences :

1. **Joke Length and Average Rating (0.62)**: There is a moderately positive correlation (0.62) between joke length and the average rating. This suggests that longer jokes tend to receive slightly higher average ratings. Users might find longer jokes more engaging or entertaining, contributing to their positive ratings.
2. **Joke Length and Rating Standard Deviation (-0.36)**: A negative correlation of -0.36 is observed between joke length and the rating standard deviation. This indicates that longer jokes tend to have less variability in their ratings. Longer jokes may have a more consistent appeal to a broader audience, leading to lower rating variability.
3. **Joke Length and Number of Users That Rated That Joke (0.15)**: The correlation between joke length and the number of users who rated the joke is positive but relatively weak (0.15). This implies that joke length has a limited influence on the number of users who rate a joke. Longer jokes do not significantly attract more users for rating.
4. **Average Rating and Rating Standard Deviation (-0.54)**: There is a strong negative correlation (0.54) between the average rating and the rating standard deviation. This suggests that jokes with higher average ratings tend to have lower rating variability. Jokes that receive high average ratings are likely to have a consistent appeal among users.
5. **Rating Standard Deviation and Number of Users That Rated That Joke (-0.41)**: A moderately negative correlation of -0.41 is observed between the rating standard deviation and the number of users who rated the joke. Jokes with a higher number of ratings tend to have lower rating variability. A larger and more diverse audience may contribute to a more consistent perception of joke quality.

5 Methodology

5.1 Data Preprocessing

5.1.1 Text Cleaning

The first step in data preprocessing involves cleaning the text data. This is achieved through the following steps:

- **Punctuation Removal** : The **clean_joke** function is implemented to clean text data, converting it to lowercase and removing punctuation and special characters. Specific replacements are made for certain patterns, and the resulting cleaned text is printed for verification.

5.1.2 Tokenization

Tokenization is a crucial step in preparing text data for machine learning models. The process involves:

- The **generate_N_grams** function is employed to tokenize and generate n-grams from the cleaned text.
- NLTK library is used for stopword removal during tokenization.
- The resulting n-grams are stored in the 'Tokens' column of the DataFrame.

5.2 Machine Learning

5.2.1 Train-Test Split

To facilitate model training and evaluation, the dataset is divided into training and testing sets using the **train_test_split** function. This ensures the model's performance is assessed on unseen data.

5.2.2 Count Vectorization

Text data is converted into a document-term matrix using the CountVectorizer technique. The resulting matrices, denoted as **X_train** and **X_test**, capture the frequency of terms within the text.

5.2.3 Linear Regression Model

In the initial phase of our analysis, we employ a linear regression model as the foundational model for predicting the target variable. The training process involves leveraging the vectorized training data, denoted as **X_train**, along with the corresponding labels **Y_train**.The linear regression model learns the relationships between the features and the target variable, aiming to capture the underlying patterns in the training data.

Subsequently, predictions are generated on the test data (**X_test**), representing a set of unseen instances. The model's performance is then rigorously evaluated using key metrics, including the R^2 score and Root Mean Squared Error (RMSE). The R score provides insights into the proportion of variance in the target variable explained by the linear regression model, with higher values indicating a better fit. Simultaneously, the RMSE offers a measure of the average magnitude of errors between the predicted and actual values, providing a comprehensive assessment of the model's predictive accuracy. These evaluation metrics play a pivotal role in gauging the effectiveness of the linear regression model in capturing the nuances of the data and making accurate predictions.

5.2.4 Lasso Regression Model

In the Lasso regression phase, we systematically train models using various alpha values to explore the impact of regularization on our predictive model. The alpha parameter acts as a tuning knob, allowing us to control the degree of regularization applied to the model. Training Lasso regression models with different alpha values helps us understand the trade-off between model complexity and predictive performance.

Following the training process, the models undergo thorough evaluation on both the training and test sets. This comprehensive assessment enables us to scrutinize how well each model generalizes to new, unseen data. By evaluating on both sets, we gain insights into potential overfitting or underfitting issues and can make informed decisions about the optimal alpha value.

In addition to standard evaluation metrics, we delve into model complexity by calculating specific metrics related to the L1-norm of coefficients and the sum of coefficients. These metrics offer a quantitative measure of the impact of regularization on the model's parameters. The L1-norm, representing the sum of the absolute values of the coefficients, and the sum of coefficients provide valuable information about the sparsity induced by the Lasso regularization, shedding light on the features considered most important by the model.

The combination of alpha tuning, comprehensive evaluation, and the analysis of model complexity metrics enhances our understanding of how Lasso regression operates under different regularization strengths. This knowledge is essential for selecting an optimal alpha value that balances model simplicity and predictive accuracy.

5.2.5 Ensemble Model

To enhance the predictive performance and robustness of our model, we employ an ensemble approach by creating a VotingRegressor. This ensemble model combines the predictive capabilities of three distinct models: Lasso Regression, Random Forest, and XGBoost. The inclusion of diverse algorithms in the ensemble allows for a complementary exploration of different modeling approaches, leveraging their respective strengths to improve overall predictive accuracy.

The ensemble model undergoes a two-fold process. Initially, it is trained on the training data, incorporating the insights and patterns learned by each individual model. The combination of predictions from these models contributes to the collective decision-making of the ensemble. The training phase aims to create a unified model that harnesses the diversity of the constituent models for improved generalization to unseen data.

Subsequently, the ensemble model is rigorously evaluated on the test data. This evaluation process assesses how well the ensemble generalizes to new, unseen instances, offering a holistic view of its predictive performance. The ensemble leverages the strengths of each constituent model to mitigate individual weaknesses, providing a more robust and reliable predictive framework.

In summary, the creation and training of the VotingRegressor ensemble model represent a strategic approach to capitalize on the complementary strengths of Lasso Regression, Random Forest, and

XGBoost. This ensemble methodology aims to achieve a superior predictive performance compared to individual models by combining their diverse perspectives and learning patterns.

5.3 Model Training

5.3.1 Support Vector Reggression

In the Support Vector Regression (SVR) phase of our analysis, we employed a **radial basis function (RBF) kernel to train the SVR model** on the vectorized training data. The chosen hyperparameters for this model were set to **c=100**, γ**=0.1**, and ϵ**=0.1**. During the training process, the SVR model learned to capture complex relationships within the data, particularly when non-linear patterns were present.

Upon evaluating the SVR model on the test set, we calculated the **Mean Squared Error (MSE) as 3.31**, providing insights into the average squared differences between the predicted and actual values. Additionally, the R score, a measure of the model's predictive performance, is yet to be added to this section after reviewing the actual output. The R score will furnish a comprehensive assessment of how well the SVR model explains the variability in the target variable, with a higher score indicating better predictive capability. These results contribute to our understanding of the SVR model's effectiveness in capturing the underlying patterns in the data and its potential utility in our regression analysis.

5.3.2 XG Boost Regression

In the XGBoost Regression phase of our study, the model underwent training on the vectorized training data, which included n-grams generated using CountVectorizer. The utilization of n-grams allows the model to capture complex patterns within the textual features. Throughout the training process, default hyperparameters were employed to ensure a baseline understanding of the model's performance.

For the evaluation phase, we scrutinized the predictive capacity of the XGBoost Regression model using key metrics. The R2 score, measuring the proportion of variance in the target variable explained by the model, was determined to be **0.02**. This relatively low R score suggests that the model struggled to account for the variability in the response variable.

Additionally, **the Root Mean Squared Error (RMSE) was calculated, resulting in a value of 1.49**. The RMSE provides a measure of the average magnitude of errors between predicted and actual values, with lower values indicating better predictive accuracy. These evaluation metrics contribute to our overall assessment of the XGBoost Regression model's performance and its suitability for our regression analysis.

5.3.3 Random Forest Regression

We trained **the model on vectorized training data, incorporating text representations obtained through CountVectorizer**, in order to investigate RandomForest Regression. With the use of this technique, the model was able to analyse and incorporate the structural information present in the n-grams that were generated from the text. We chose a particular hyperparameter configuration for the training phase, exactly setting the number of estimators to 100. This choice was made in an effort to strike a balance between computational efficiency and model complexity. After training was finished, we used basic metrics to evaluate the RandomForest Regression model's performance. After computation, it was found that the target variable's R score, which indicates the percentage of variance explained by the model, was 0.02. Higher scores indicate more robust predictive capabilities. This score gives an insight into the model's ability to identify patterns in the data.

Furthermore, the **Root Mean Squared Error (RMSE)** was computed, **yielding a value of 1.22**. Lower values indicate improved predictive accuracy. The Root Mean Square Error (RMSE) serves as a barometer for the average magnitude of errors between predicted and actual values. These assessment metrics add to our comprehensive assessment of the RandomForest Regression model's functionality and suitability for our regression analysis.

6 Conclusion

The comparison of the linear regression model and the ensemble model reveals a similarity in their R scores and Root Mean Squared Error (RMSE) values. This suggests that, based on the selected evaluation metrics, both models perform similarly in capturing the underlying patterns within the data and making accurate predictions. Further analysis of other metrics and model characteristics may provide additional insights into their relative strengths and weaknesses.

The utilization of Lasso regression for regularization introduces a tuning parameter, alpha, which plays a crucial role in controlling the model's complexity. The choice of alpha not only impacts the regularization strength but also influences the overall performance of the model. Adjusting alpha allows for a trade-off between model simplicity and its ability to fit the training data, and this balance is crucial for achieving optimal predictive accuracy.

To enhance predictive accuracy, it is advisable to explore additional avenues such as feature engineering or experimenting with different algorithms. Feature engineering involves manipulating or creating new features from existing ones to provide the model with more relevant information. Additionally, trying alternative machine learning algorithms may offer different perspectives on the data and potentially lead to improved model performance.

In summary, the project's predictive accuracy could be enhanced through a thoughtful exploration of feature engineering techniques and the consideration of alternative algorithms beyond the linear regression and ensemble models. These steps can contribute to a more comprehensive understanding of the data and potentially lead to better predictions.

References

https://medium.com/mitb-for-all/rating-prediction-from-review-text-with-regularization-linear-regression-vs-logistic-regression-df0181fe9c07

https://towardsdatascience.com/user-user-collaborative-filtering-for-jokes-recommendation-b6b1e4ec8642

Tang Y., Liao D., Huang S., Fan Q., and Liu L., "Construction of Machine Learning Model Based on Text Mining and Ranking of Meituan Merchants," Scientific Programming, vol. 2021, 9 pages, 2021. 10.1155/2021/5165115 5165115