Jokes Recommendation system with Jester Dataset

Team 21:
Animesh Pareek (2021131)
Sameer Gupta (2021093)
Gunjapalli Sravani Reddy (MT22098)
Abhijay Tiwari (2021439)

Abstract

Dataset: The Jester Dataset includes 100 jokes, and ratings for these jokes on a scale of (-10, 10), provided by 73,421 users. (Total ratings: 4.1 Million)
Aim: Build a joke recommendation system using classical machine learning techniques, that given a joke, predicts its rating.

---

1 Exploratory Data Analysis

This sections includes, all the EDA performed on the Datasets provided.
"Exploratory Data Analysis (EDA) is an initial phase in data analysis, involving the use of statistical methods and visualizations to understand patterns, relationships, and characteristics within a dataset."

1.1 Jokes Dataframe

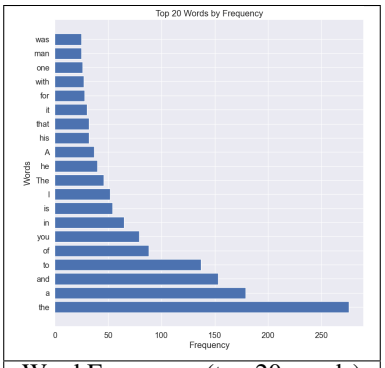
This Dataframe involves all 100 jokes provided in the dataset.
Columns: JokeID, Joke

joke_id	joke
1	A man visits the doctor. The doctor says "I have bad news for you.You havecancer and Alzheimer's disease". The man replies "Well,thank God I don't have cancer!"

Jokes

Table 1: Jokes Dataframe

- The original format of the given Jokes Dataframe was 100 HTML files with each file containing one joke.
- We merged them together to create a common dataframe containing (JokeID, joke).
- For each joke, we encoded the words present, and then calculated the frequency of each word in the entire dataset.
- Common words such as articles were identified, as those would not be useful in any of the henceforth predictions.



Word Frequency (top 20 words)

```
common_words = ["---","---|---","i", "me", "my", "myself", "we", "our", "ours", "ourselves", "you", "your", "yours", "yourself", "yourselves",
```

Common words

Table 2: EDA on jokes dataframe

1.2 Ratings Dataframe

This Dataframe involves all ratings provided by each user, for all jokes.
Columns: UserID, Joke1, Joke2, Joke3, etc.

user_id	number_o	joke_1	joke_2	joke_3	joke_4	joke_5	joke_6	joke_7	joke_8	joke_9	joke_10	joke_11	joke_12	joke_13	joke_14	joke_15	joke_16	joke_17	joke_18	joke_19	joke_20	joke_21
1	74	-7.82	8.79	-9.66	-8.16	-7.52	-8.5	-9.85	4.17	-8.98	-4.76	-8.5	-6.75	-7.18	8.45	-7.18	-7.52	-7.43	-9.81	-9.85	-9.85	-9.37

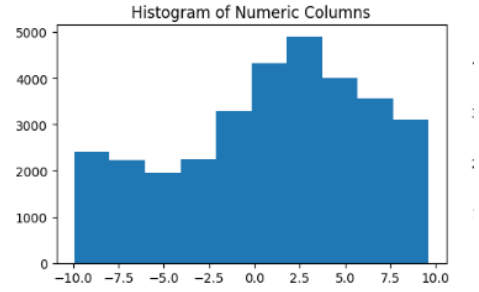
Ratings

Table 3: Rating Dataframe

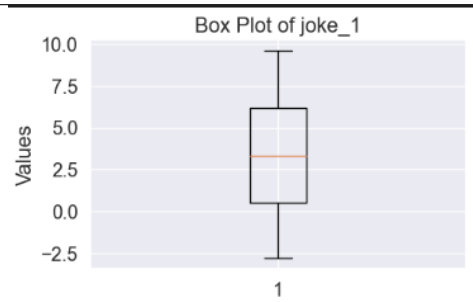
- The original format of the given ratings dataframe was as follows:
  - jesterdataset11.zip: (3.9MB) Data from 24,983 users who have rated 36 or more jokes, a matrix with dimensions 24983 X 101.
  - jesterdataset12.zip: (3.6MB) Data from 23,500 users who have rated 36 or more jokes, a matrix with dimensions 23500 X 101.
  - jesterdataset13.zip: (2.1MB) Data from 24,938 users who have rated between 15 and 35 jokes, a matrix with dimensions 24,938 X 101.
- We merged them together to create a common dataframe containing (userID, rating for each joke (joke1, joke2, joke3, etc)).
- For each joke, we calculated, the description factors, to identify the range of ratings present for each joke. (non normalized)
- For each joke, we calculated, the description factors, to identify the range of ratings present for each joke. (normalized)
- We identified sparse (users with any joke unrated) and dense users (users with all jokes rated).
- Change in observations based on NAN approximations, by:
  - Replacing NAN values with 0.
  - Replacing NAN values with mean rating.

	user_id	number_o	joke_1	joke_2	joke_3	joke_4	joke_5	joke_6	joke_7	joke_8	joke_9	joke_10	joke_11	joke_12	joke_13	joke_14	joke_15	joke_16	joke_17	joke_18	joke_19	joke_20
count	73421	73421	32024	35277	32402	30512	73405	39599	73401	73411	30741	39352	42677	43564	73405	43899	73406	73399	73413	73404	73406	73410
mean	36711	56.33756	0.901997	0.162989	0.193411	-1.4126	0.235352	1.330277	-0.63395	-0.99627	-0.62004	1.181706	1.728665	1.50436	-1.94486	1.320285	-2.11251	-3.10314	-1.2881	-0.90616	0.17156	-1.20327
std	21194.96	29.01569	5.242998	5.627972	5.448998	5.276202	5.302902	4.976994	5.610902	4.968263	5.201667	5.183989	4.942995	4.843209	5.137793	5.136872	5.123141	5.02833	4.439184	4.993185	5.048561	5.158966
min	1	15	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95	-9.95
25%	18356	29	-2.82	-4.51	-4.22	-5.97	-4.03	-1.75	-5.87	-5.19	-4.85	-2.38	-1.31	-1.55	-6.55	-2.23	-6.6	-7.48	-4.71	-4.95	-3.69	-5.58
50%	36711	52	1.6	0.73	0.68	-1.41	0.92	1.89	-0.29	-0.63	-0.24	1.84	2.38	2.09	-2.14	1.99	-2.545	-3.88	-0.73	-0.49	0.63	-0.97
75%	55066	75	5.05	4.85	4.56	2.57	4.37	5.24	3.88	2.77	3.3	5.34	5.73	5.34	2.14	5.49	1.89	0.63	1.8	2.86	3.98	2.72
max	73421	100	9.61	9.37	9.61	9.37	9.51	9.66	10	9.76	9.9	9.37	9.61	9.85	9.76	9.81	9.9	9.37	9.81	9.81	10	9.71

Non normalized Ratings Description



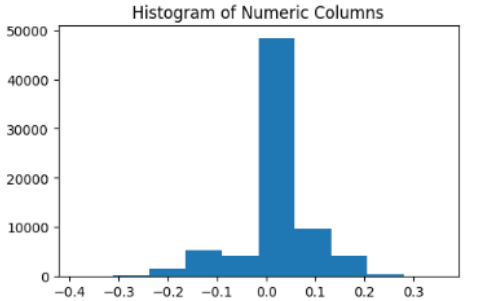
Histogram



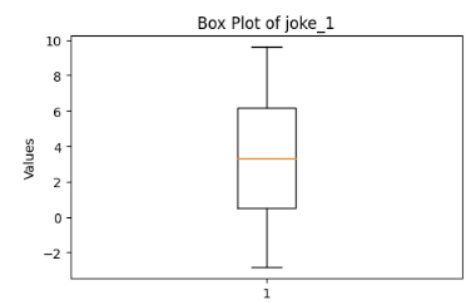
Box Plot

	user_id	number_o	joke_1	joke_2	joke_3	joke_4	joke_5	joke_6	joke_7	joke_8	joke_9	joke_10	joke_11	joke_12	joke_13	joke_14	joke_15	joke_16	joke_17	joke_18	joke_19	joke_20
count	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421	73421
mean	36711	56.33756	0.008442	0.001295	0.001956	-0.01278	0.00432	0.016076	-0.02104	-0.03308	-0.00531	0.014524	0.022949	0.020297	-0.05851	0.017717	-0.0636	-0.09356	-0.04104	-0.02862	0.003123	-0.03848
std	21194.96	29.01569	0.071322	0.082276	0.073752	0.06956	0.15802	0.078509	0.165999	0.151423	0.067876	0.08263	0.084822	0.083221	0.156491	0.088711	0.155255	0.154685	0.135913	0.148328	0.148093	0.151598
min	1	15	-0.38609	-0.54691	-0.47272	-0.4363	-0.75572	-0.40597	-0.70322	-0.76852	-0.39362	-0.44029	-0.46754	-0.42714	-0.95327	-0.5709	-0.95116	-0.9925	-0.9104	-0.65656	-0.83644	-0.77212
25%	18356	29	0	0	0	0	-0.11679	0	-0.15449	-0.14376	0	0	0	0	-0.17313	0	-0.1767	-0.20246	-0.13363	-0.13991	-0.10711	-0.15282
50%	36711	52	0	0	0	0	0.02683	0	-0.00734	-0.01877	0	0	0	0	-0.06174	0	-0.07444	-0.11137	-0.02407	-0.01421	0.01862	-0.03183
75%	55066	75	0.013361	0.009875	0	0	0.120523	0.059416	0.10969	0.078813	0	0.058267	0.081736	0.077243	0.061713	0.078581	0.054203	0.018576	0.050415	0.081016	0.110179	0.07757
max	73421	100	0.354335	0.438057	0.343277	0.322166	0.684337	0.407074	0.743273	0.632609	0.293553	0.440363	0.496522	0.403345	0.745227	0.464431	0.670765	0.811077	0.762075	0.557612	0.807356	0.539417

Normalized Ratings Description



Histogram



Box Plot

user_id	number_o	joke_1	joke_2	joke_3	joke_4	joke_5	joke_6	joke_7	joke_8	joke_9	joke_10	joke_11	joke_12	joke_13	joke_14	joke_15	joke_16	joke_17	joke_18	joke_19	joke_20	joke_21
1	74	-7.82	8.79	-9.66	-8.16	-7.52	-8.5	-9.85	4.17	-8.98	-4.76	-8.5	-6.75	-7.18	8.45	-7.18	-7.52	-7.43	-9.81	-9.85	-9.85	-9.37
2	100	4.08	-0.29	6.36	4.37	-2.38	-9.66	-0.73	-5.34	8.88	9.22	6.75	8.64	4.42	7.43	4.56	-0.97	4.66	-0.68	3.3	-1.21	0.87

Users (sparse and Dense)

Table 4: EDA on rating dataframe

2 Methodology

"Methodology is the structured approach used to gather, interpret, predict, and draw conclusions from data or information."

2.1 Operations on Jokes Dataframe:

After forming our merged dataset, we prepossessed it in order to use in order to use it in model training.

joke_id	joke
1	A man visits the doctor. The doctor says "I have bad news for you.You havecancer and Alzheimer's disease". The man replies "Well,thank God I don't have cancer!"

Jokes

Table 5: Rating Dataframe

2.1.1 Preprocessing

Preprocessing included the following steps:

- Cleaning out common words from the jokes.
- For each of the unique words, lemmentize the word (using NLTK.stem), filter the words based on POS tagging (using NLTK.tokenize, NLTK.postag) to obtain only NOUNS and VERBS, and then calculating the probability of each word being the "main topic" for that joke (using LDA).
- Once (topic, probability) is obtained, we can store that in the dataframe for further use.

joke_id	joke	Processed_joke	cluster	sorted_topics	main_topic
1	A man visits the doctor. Th	man visit doctor doctor say news ha	[(0, 0.002553701), (1, 0.0021722557), (2, 0.0021	[(9, 0.9774125), (4, 0.002951629), (7, 0.0028713304), (6, 0.0026840565), (0, 0.002553701), (3, 0.0025248607	9

Jokes

Table 6: Rating Dataframe

2.1.2 Normalization

No normalization for required for the Jokes Dataframe.

2.2 Operations on Ratings Dataframe:

This Dataframe involves all ratings provided by each user, for all jokes.
Columns: UserID, Joke1, Joke2, Joke3, etc.

user_id	number_o	joke_1	joke_2	joke_3	joke_4	joke_5	joke_6	joke_7	joke_8	joke_9	joke_10	joke_11	joke_12	joke_13	joke_14	joke_15	joke_16	joke_17	joke_18	joke_19	joke_20	joke_21
1	74	-7.82	8.79	-9.66	-8.16	-7.52	-8.5	-9.85	4.17	-8.98	-4.76	-8.5	-6.75	-7.18	8.45	-7.18	-7.52	-7.43	-9.81	-9.85	-9.85	-9.37

Ratings

Table 7: Rating Dataframe

2.2.1 Preprocessing

Preprocessing included the following steps:

- Using our earlier observations, we prefilled the NAN ratings with 0.
- Included a column "Number of jokes rated by user", this helps keep track of sparse and dense users.

2.2.2 Normalization

Normalization of the ratings was done to test if the results from normalizing the data were better.
Conclusion: It was found that normalization does not help in prediction.

user_id	number_o	joke_1	joke_2	joke_3	joke_4	joke_5	joke_6	joke_7	joke_8	joke_9	joke_10	joke_11	joke_12	joke_13	joke_14	joke_15	joke_16	joke_17	joke_18	joke_19	joke_20	joke_21
1	74	-0.12541	0.140966	-0.15492	-0.13086	-0.1206	-0.13632	-0.15797	0.066875	-0.14401	-0.07634	-0.13632	-0.10825	-0.11515	0.135513	-0.11515	-0.1206	-0.11916	-0.15732	-0.15797	-0.15797	-0.15027

Normalized Ratings

Table 8: Normalized Rating Dataframe

2.3 Model Training:

In model training, we discovered various ways of prediction, including prediction of ratings for sparse users, recommending jokes to users based on their interests, and predicting ratings for jokes based on prior user ratings.

2.3.1 Rating Prediction

This includes predicting a cumulative ratings based on all user for a particular joke.
Steps Involved:

- Given a joke, we preprocess it.
- As we did previously, we take out the topics probable for the joke.
- For each topic, we calculate its correlation with every other joke, using pearson coefficient.
- We pick all jokes that have correlation higher than a threshold.
- For each joke we calculate a combined rating, using mean ratings and correlation factor.

This accounts for our final rating for the joke.

2.3.2 User-User Based

This includes predicting the rating that a user would give to a joke, based on how they had rated all the other jokes.
Steps Involved:

- We split the dataframe into sparse and dense users.
- We pick a sparse user (randomly).
- Using the dense user dataframe, we calculate the correlation between each of the dense users with the sparse user.
- Using these correlations and the ratings given to the joke by the dense users, we predict the rating that the sparse user would provide.

This accounts for our final rating for the joke by that user.

```
user_id                68388.000000
number_of_jokes_rated     72.000000
joke_1                     2.714499
joke_2                     1.398703
joke_3                     2.590480
                           ...
joke_96                    3.045859
joke_97                    4.111090
joke_98                    2.110463
joke_99                    1.942481
joke_100                   2.579359
```

Predictive ratings for sparse user

Table 9: Results of User User Based

2.3.3 Joke Content Based

This includes recommending 20 jokes from the prior joke dataset, for a user that has not rated that joke previously.
Steps Involved:

- Using the (topic, probability) sets found for each joke, we pick the topic which is the most likely (maximum probability).
- For the sparse user chosen, we determine the interested topics by that user, using their rating history.
- For these interested topics, we collect the jokes with those topics as their "main topics".
- If the number of jokes collected is less than required, we go for another topic.
- Once these are collected, we sort them according to most favoured and least favoured.

This accounts for our joke recommendations for a particular user.

Recommending Jokes for user 68388 Joke 1: A couple has been married for 75 years. For the husband's 95thbirthday, his wife decides to surprise him by hiring a prostitute.That day,
Recommended Jokes with LDA

Table 10: Results of Content Based

---

3 Results:

Our Model aims at providing a detailed recommendation for users that have not rated a joke, along with a cumulative rating for any new joke, based on user feedback from the training sets.

Exploring the dataset we found multiple ways of prediction including rating prediction for a joke, or for a user, and recommending jokes to a user. These have all been implemented in the project.

---

4 Existing Works:

Surgoku

- Using Deep Learning Algorithms, Factorizaion is performed on the dataset.
- These Factorizations include, Rank, Rating, and Item.

Abbi163

- Setting up SQL to store the databases.
- Uses UUCL and Content Based (partially) to provide recommendations.
- Setting GUI.

Junolee

- This repository using tools included in GraphLab to evaluate and build the recommnder system.
- It includes Collaborative filtering with the following steps.
  - Data Exploration
  - Choice of model training

  - Parameter tuning
  - Model results and optimization

JoeDockrill

- This repository using FastAI collaborative filtering model to recommend jokes from the dataset.
- An environment is set up for data preprocessing.
- Building the collaberative model using FastAI.
- Testing those predictions using both batch learning and indiviual testing.
- GUI setup.

4.1 How is ours different:

The existing works, primarily rely a lot on Deep Learning, and are either incomplete or unexplained. Our work however, includes only classical machine learning methods, while having several detailed results based on user choice, and includes feedback taken from user's rating history, or from other related users.

---

5 Conclusions and Key Learning's:

Some of our key learning from the project was on the prepossessing techniques used in text and numerical based data.
We learnt how to lemmentize texts, tokenize them, POS tag them, and figure out primary words from a line of texts.
We learnt about collaborative filtering techniques and what types of recommendation systems and data for them are present, and how the methodology for each differs.
