Title:

Joke Recommendation System

Team Group - JRS

Amruta Dhondage

Navneet Jain

Surbhi Jain

Project Description:

We propose to build a joke recommendation system based on user's preferences using the data obtained using UC Berkeley's Jester Portal. Jester has a collection of hundreds of jokes within millions of ratings provided by hundreds and thousands of users. We aim to a model which understand user's preference, likes/dislikes to recommend the most engaging jokes for the user. We further aim to analyze the categorization of jokes into multiple domains. For recommendation purpose, we will try to find the nearest domain and then predict the jokes from the closest domain for a given user.

Secondly, we can categorize the jokes into buckets and use the user ratings data to place every user in a particular category of jokes. Each user can like more than one category of jokes. By this we can find a target audience for a new joke in the market. The approach will be, place the new joke in one of the buckets and the send it to the users in that bucket.

Objectives:

1. Build a recommendation system which looks for user's preferences
2. Perform categorization of jokes
3. Mapping the users to nearest category of jokes which user might be most interested in.
4. Finding the target audience for each new joke in the market. Like this new jokes will only be sent to the audience who are most likely to read them in comparison to sending to every user.

Dataset:

- jester_dataset_1_1.zip: (3.9MB) Data from 24,983 users who have rated 36 or more jokes, a matrix with dimensions 24983 X 101
- Ratings are real values ranging from -10.00 to +10.00 (the value "99" corresponds to "null" = "not rated").
- The first column gives the number of jokes rated by that user. The next 100 columns give the ratings for jokes 01 - 100.
- The text of the jokes can be downloaded here: jester_dataset_1_joke_texts.zip (92KB)

Evaluation:

We will perform n-fold cross validation on the data available. We will split the data in training and test buckets and then perform 10-fold cross-validation. We will try to minimize the RMSE if we solve it in a regression setting. In a classification setting we will try to maximize: Accuracy, Precision, Recall and F-score. We will also try to compare our results with standard benchmarks.