04/28/17
James Delorey delorey2
Siva Mullapudi scm5

Movie Rating Project

Introduction

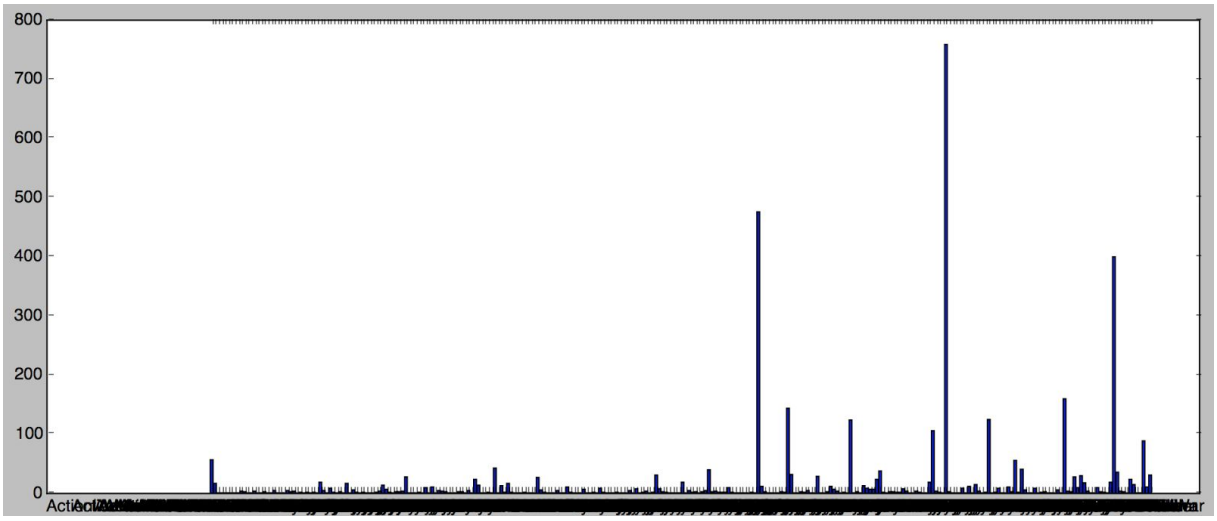
The goal of the Movie Rating Project is to implement an algorithm, such that given a user and a movie, the procedure can predict the corresponding rating that the user would assign to a movie. Such an algorithm is vital to businesses like Netflix or Google Play because their business model depends on their ability to increase engagement, and thus revenue, per customer by recommending content customized to each user. Creating a successful ratings prediction algorithm proved difficult for a number of reasons. Firstly, a sizable portion of the dataset was noisy and incomplete, which can bog down some classifiers. In addition, the training data was quite sparse when compared to the overall set of combinations of users and movies, which makes finding common patterns challenging. To avoid this, we have opted to use a Stochastic Gradient Descent algorithm with bias correction, which was robust to the noisy and missing data in training and performed well to predict new ratings.

Project Description

The problem

The Movielens dataset consists of 3,883 movies, 6,040 users, and 802,553 ratings. Given this training set, our goal is to generate an algorithm that can reliably predict a user's rating of a previously unseen movie.

Of the 3885 total movies, roughly 10% have null values for Year and Genre respectively. And of the 6041 total users, roughly 10% have null values for Gender, Age, and Occupation respectively. The Genre feature also poses a difficult problem because each movie may belong to multiple genres. The feature is constructed of multiple genres joined together by a pipe character, and the cardinality, shown below, is prohibitively high.



Our early exploration included testing five different stock classification algorithms for their accuracy against the unprocessed dataset. The non-cross-validated tests below were conducted by evaluating the classifier on a 80/20 split of the training data. The cross-validated

tests use 5 folds of the training data to estimate the accuracy.The results of our initial suite of tests are as follows:

- Accuracy of Decision Tree classifier: 0.32952881734
- Accuracy of Random Forest classifier: 0.338313262019
- Accuracy of Gaussian Naive Bayes classifier: 0.286161073073
- Accuracy of Bernoulli Naive Bayes classifier: 0.348524400197
- Accuracy of K Nearest Neighbors classifier: 0.317990667306
- Cross-Validated Accuracy of Decision Tree: 0.32 (+/- 0.00)
- Cross-Validated Accuracy of Random Forest: 0.33 (+/- 0.00)
- Cross-Validated Accuracy of Gaussian Naive Bayes: 0.29 (+/- 0.01)
- Cross-Validated Accuracy of Bernoulli Naive Bayes: 0.35 (+/- 0.00)
- Cross-Validated Accuracy of K Nearest Neighbors: 0.31 (+/- 0.00)

Our initial thought was that the sparsity in the data was significantly affecting the accuracy of our classification algorithms. There are many techniques to handle missing data such as, ignoring the tuple, treating null values as a distinct feature value, or replacing null values with a generated value. Different techniques for generating missing values include using a measure of central tendency, using a measure of central tendency from all samples belonging to the same class, or using the most probable value based on some type of regression, Bayesian formalism, or decision tree. We attempted each one of these methods, as well as normalizing the year attribute, and the best result improved our baseline implementation by only 0.11%. We concluded that traditional classification algorithms were not ideal for solving this problem.

Prior Work

Ultimately we recognized that the rating problem was similar to recommender systems common in commercial domains like movies, restaurants, social networks, and music. These recommender systems are mainly composed of two approaches, content based filtering and collaborative filtering. The classifiers mentioned above employ a content based filtering, since they compare the user and movie features to generate predictions. On the other hand, collaborative filtering algorithms build a model based on a user's past ratings and how they perceived different types of movies, which in turn is used to predict a user's future ratings (Aberger, Christopher R. "Recommender: An Analysis of Collaborative Filtering Techniques.").

Collaborative filtering generates a prediction from two values. Each rating a user makes is used to generate a score for that user. Each prediction for a given movie across the entire population is also considered to generate a score for that movie. In this way, collaborative filtering algorithms take into account both the user's tendencies and the intrinsic value of movie as noted by the population's overall perception. (Schafer, J. H. J. B., et al. "Collaborative filtering recommender systems." The adaptive web (2007): 291-324.)

Proposed Model

Our new implementation, a bias stochastic gradient descent algorithm (B-SGD), is a collaborative filtering algorithm and therefore, circumvents the sparsity of many of the attributes in the user and movie datasets.

To initiate the bias-SGD procedure, one needs to construct the rating matrix and its matrix factors. The user ratings matrix, r, has dimensions |u| by |m| where $r_{i,j}$ consists of the rating for movie *j* given by user *i* and 0 otherwise. With the majority of the matrix being empty, since a user will only rate a few movies, the goal is to fill in the rest of the matrix with values that best reflect how users will rate movies that they have not reviewed yet.

This can be accomplished by discovering the latent features that determine how a user determines a rating for a movie. However, given the considerable amount of users and movies present in the movielens data, the quantity of latent features can be incredibly large and can cause problems for training time. The matrix factorization technique provides an elegant solution, as it reduces the dimensions of the ratings matrix into a product of two latent matrices, p (user data) and q (movie data).

$$r = p * q$$ where p is (|u| by k) and q is (k by |m|)

The row p_i consists of the features for the user i while column q_j contains those of the movie j. To generate more accurate results, the bias exerted by a user, b_i, for a movie, b_j, and the overall rating average, μ, are taken into consideration as well. This means that a rating r_{ij}, given by user i for movie j, can be computed as follows:

$$r_{ij} = p_i * q_j + \mu + b_i + b_j$$

In order to compute the best factorization, we seek to minimize this error:

$$min \quad \sum_{i,j} (r_{ij} - b_i - b_j - (p_i * q_j))^2 + \lambda(||p_i||^2 + ||q_j||^2 + {b_i}^2 + {b_j}^2)$$

An L2 regularization term is added to the error term in order to avoid overfitting.

By first calculating the determinants for all of the variable above and applying a SGD approximation, we end up the following equations:

$$e_{ij} = r_{ij} - (p_i * q_j)$$
$$b_i \approx b_i + \eta(e_{ij} - \lambda b_i)$$
$$b_j \approx b_j + \eta(e_{ij} - \lambda b_j)$$
$$p_i \approx p_i + \eta((e_{ij} * q_j) - \lambda p_i)$$
$$q_j \approx q_j + \eta((e_{ij} * p_i) - \lambda q_j)$$

where η denotes the step length while λ is the regularization constant

We can now fill in the values for the factor matrices as we apply the equations above by repeatedly iterating through the non-zero entries for the ratings matrix. We determined that the optimal accuracies occurred in using a step length of 0.01, procedure length of 50 steps, lambda with 0.06 and the number of latent factors of 50.

04/28/17
James Delorey delorey2
Siva Mullapudi scm5

Results

At the midterm checkpoint our highest accuracy was 0.34968 using a Bernoulli naive bayes classifier and handling missing values by replacing them with the value of the most similar tuple. Basic stochastic gradient descent improved that score to 0.43053. By adding bias correction and tweaking some parameters, we increased that accuracy to 0.45795.

Collaborative filtering in the form of bias stochastic gradient descent showed significant improvement over classification algorithms. We assume that this improvement is due to the fact that B-SGD relies on tendencies of certain users as well as the perception of the entire population to predict ratings and therefore is not susceptible to the noisy attributes in the training data.

Work Distribution: James - 50%, Siva - 50%