Kira Mathias-Prabhu
kmathias
April 26 2017

10-601 HW9: Recommendation Systems

Problem Statement: I chose to develop a recommendation system for the Movie Lens dataset using Matrix Factorization. The dataset provided movie ratings from 1 to 5 by User ID and Movie ID. The goal was to take this rating data and use it to generate user and movie feature vectors in k-dimensional space which could be used to predict user ratings for movies, i.e. to be able to fill in the 0 entries in the ratings matrix.

Data Pre-processing and Feature Engineering: The ratings data available in 'training_ratings.dat' was the only part of the dataset used directly to train the model. The data was "`::`"-separated into three fields: `User ID, Movie ID, Rating`. So the first preprocessing step was to split this into an N x 3 array, where N = number of ratings. However, some entries were missing fields, so as part of the preprocessing, these had to be deleted. The final preprocessing step was to rearrange the rating matrix `R` so that User IDs were rows, Movie IDs were columns, and the entry at `R(i,j)` = User i's rating of Movie j.

To actually train the model, matrices U representing the users in R, and V representing the movies in R were generated. If R was size m x n, U had size m x k and V had size n x k. Thus each user was represented by a k-dimensional feature vector in U, and each movie was represented by a k-dimensional feature vector in V. The value of k was chosen to be 10 (see next section for more details on parameter and hyper-parameter selection). A rating by user i of movie j could then be characterized by the dot product of the i^{th} row of U with the j^{th} row of V. Note that bias terms were folded into the feature vectors as the 0^{th} entry of each feature.

Model Implemented (including choice of parameters): I implemented a Matrix Factorization model with regularization and bias, which was trained using Stochastic Gradient Descent. There were two bias vectors - one for the user matrix U, and one for the movie matrix V. Thus, there were 4 parameters I had to tune: k, learning rate, regularization for U, regularization for V.
In order to select parameters, I partitioned the dataset into a train and validation set, where the validation set consisted of a random sampling of ratings from each user. I originally randomly partitioned the dataset to get a validation set, but found that ensuring there were ratings from every user in the test set made it more realistic, and therefore more useful in evaluating the accuracy of the model.

I then tuned my parameters one by one, starting with k (and no bias terms). Once I had chosen a k, I added in bias terms, and tuned regularization parameters, and finally learning rate. The results of my benchmarking are summarized below. The parameter values used in the final submitted model are bolded. Additional values were tested, but only three have been selected to illustrate the range of RMSE observed below:

Parameter Being Tuned Other param values (held fixed)			
k No bias, Learning rate = 0.001, Ureg = Vreg = 0.1	**k=10** RMSE:1.06	k=45 RMSE:1.68	k=78 RMSE:2.45
Ureg, Vreg Learning rate = 0.001, k=10	**Ureg=0.1, Vreg=0.1** RMSE:1.068	Ureg=1/|U|, Vreg=1/|V| RMSE:1.156	Ureg=0.15, Vreg=0.15 RMSE:1.0644
Learning Rate k=10 Ureg=Vreg=0.1	LR = 0.002 RMSE:0.899	LR = 0.005 RMSE:0.859	**LR = 0.01** RMSE:0.812

I trained my model using 50 iterations of SGD on just the training set (where each iteration consisted of a full cycle through all training samples in random order) in order to select the parameters. I then used 100 iterations of SGD on the full dataset to make the predictions for the MovieLens test set.

Analysis of Results (including why metric used for analysis of task was appropriate)
The results were analyzed using the Root Mean Squared Error (RMSE) metric. For my local evaluation purposes, I computed the RMSE of my training and test sets after each iteration of SGD, and used the final RMSE of both the training and validation sets to evaluate the overall performance and generalization capabilities of my model. This was an appropriate metric because it essentially computes the average error among all rating predictions by comparing them to the true rating. My final model trained only using my training partition and tested on my validation partition with 100 iterations of SGD achieved a RMSE of 0.812 on the validation set. My final model trained on the full dataset achieved an error of 0.684 on my validation set and 0.694 on my training set, and a RMSE of 0.871 on the test set.

Collaboration: I did not give or receive help with this assignment.

Time spent: 14 hours