EECS 349 (Machine Learning) Homework 4

How to submit your homework

1. Create a **PDF document** containing answers to the homework questions. Show your reasoning in your answers.
2. Include source code for the program you write.
3. Compress all of the files specified into a .zip file.
4. Name the file in the following manner, *firstname_lastname_hw4.zip*.
3. Submit this .zip file via Canvas by the date specified on Canvas.

1) Collaborative Filtering: Looking at the Data (2 points)

When doing machine learning, it is important to have an understanding of the dataset that you will be working on. On the course website under "links" there is a link to the MovieLens dataset. You will use the MovieLens 100K dataset to build a collaborative filter to predict the rating a user may give to a movie they haven't seen. This dataset has 100,000 ratings from 943 users on 1682 movies.

A. (1 point) Go through each pair of users. For each pair of users, count how many movies both have reviewed in common. What is the mean number of movies two people have reviewed in common? What is the median number? Plot a histogram of this data where each bar is some number of movies reviewed in common and the height of the bar is the number of user pairs who have reviewed that many movies in common. Clearly label your dimensions. Explain any choices you made in the generation of this histogram.

B. (1 point) Go through the movies. For each movie, measure how many reviews it has. What movie had the most reviews? How many reviews did it have? What had the fewest? Order the movies by the number of reviews each one has. Now, plot a line where the vertical dimension is the number of reviews and the horizontal dimension is the movie's number in order by the number of reviews. Clearly label your dimensions. Now that you have this data, do you think the number of reviews per movie follows Zipf's law? (http://en.wikipedia.org/wiki/Zipf's_law)

2) Collaborative Filtering: Distance measures (1 point)

A. (1/2 point) Assume a user will be characterized by a vector that has that user's ratings for all the movies in the MovieLens data. This means each user is described by a vector of size 1682 (the number of movies). Assume the distance between two users will be their Manhattan distance. Here is the problem: most users have not rated most of the movies. Therefore, you can't use Manhattan distance until you decide how to deal with this. One approach (call it approach A) is to put a 0 in for every missing rating. Another approach is to find the average rating chosen by that user on the movies he/she DID rate. Then, substitute that value in for all missing movie ratings (call it approach B). Which approach do you think is better? Say why. Give a toy example situation (i.e. 3 users, 5 movies, a few missing ratings) that illustrates your point.

B. (1/2 point) You are trying to decide between two different distance measures for movies: Euclidean distance and one based on Pearson's correlation. Each movie is characterized by a vector of 943 (the number of users) ratings. Assume that all missing ratings values have the value 3 filled in as a place-holder. Which distance measure do you think would be better for item-based collaborative filtering? Why do you think that? Illustrate your point with a toy example. Question removed to simplify homework.

EECS 349 (Machine Learning) Homework 4

3) Collaborative Filters (4 points)

You will now build two collaborative filters, user-based and item-based collaborative filters. We have provided two callable python scripts (`user_cf.py` and `item_cf.py`) that have some starter code. You will complete the functions named `user_based_cf()` and `item_based_cf()` in the starter code. Filter *user_cf.py* must be a user-based collaborative filter and *item_cf.py* must be an item-based collaborative filter. All distances will be in the user/movie ratings space. Therefore movies will each have a 943-element vector (one element per user) and the *i*th element will contain the rating of that movie by user *i*. People will have 1682 element vectors, where the *j*th element contains that user's rating for movie *j*. Missing ratings must be filled in by the value 0. The predicted rating returned will be the mode (the number that occurs most frequently in the set) of the top K neighbors. Below is how each script will be run from the command line.

```
python user_cf.py <datafile> <userID> <movieID> <distance> <k> <i>

python item_cf.py <datafile> <userID> <movieID> <distance> <k> <i>
```

The input variables are as follows:

<datafile> - a fully specified path to a file formatted like the MovieLens100K data file *u.data*

<userID> - a userId in the MovieLens100K data

<movieID> - a movieID in the MovieLens 100K data set

<distance> - a Boolean. If set to 0, use a distance measure based on Pearson's correlation. If 1, use Manhattan distance.

<k> - The number of nearest neighbors to consider

<i> - A Boolean value. If set to 0 for user-based collaborative filtering, only users that have actual (ie non-0) ratings for the movie are considered in your top K. For item-based, use only movies that have actual ratings by the user in your top K. If set to 1, simply use the top K regardless of whether the top K contain actual or filled-in ratings.

Each call of your collaborative filters should create an output on the command line that contains the following information, in the order below, formatted as shown in the example below. In real use, the numbers here would vary depending on userID, movieID, etc.

```
> userID:123    movieID: 456    trueRating: 2    predictedRating: 4    distance:1    K:3    I:1
```

4) Test your movie ratings predictor (3 points)

Your final job is to determine the effectiveness of your collaborative filters. In our test implementation, predicting one rating took 1 second on a laptop. Therefore n-fold cross validation will be too slow. Instead, we want you to take 50 *samples* of the data, each with 100 ratings.

To do one draw from the data set, randomly select a single rating. All ratings should have an equal probability of being drawn from the data set. All draws should be done with replacement. One sample should have 100 draws from the data set. Once you've taken 50 samples, keep these samples and use them for ALL your experiments.

To do an experiment on a sample, split the data into a test set (the 100 ratings in your sample) and the prior data (the other 99,900 ratings).

For each of the 100 movie-ratings in your sample, hand the movieID-userID pair as input to a variant of your collaborative filter that uses the other 99,900 ratings as prior data and get back the predicted rating. You can then compare the predicted rating to the actual rating using an error measure you define. Averaging the error over the 100 ratings in a sample gives you the sample error.

Repeat this process for each of the 50 samples. This will give you 50 (probably, mostly) independent, identically distributed estimates of how well this variant of your collaborative filter predicts ratings.

You can then repeat this for other variants and compare their performance using an appropriate statistical test.

A. (1/2 point) Clearly define an error measure for collaborative filtering that you will use in your experiments. Use a formula, text and an example to explain your error measure. Explain why you made this choice. *(Consider the following: Should the grade on each prediction be right/wrong? Or, does the degree of error on each rating matter? Maybe predicting a 4 when the truth is 1 is worse than predicting a 2 when the truth is 1…or maybe not. Maybe wrong is just wrong.)*

B. (1/2 point) Define the statistical test you will use in subsequent experiments to determine whether the difference between two system variants is statistically significant. Explain why you chose the test you selected. What is the null hypothesis? What level of confidence (p value) have you decided is sufficient to call the results from two filters statistically different?

For Parts C, D, E, F you are required to explain the assumptions you made in answering each question and state how you set the parameters you did NOT vary when answering that particular question. You must show a graph for each one that illustrates the difference (or lack of difference) between your variants. You must say whether the difference is statistically meaningful and back that up with the results of your statistical test(s).

C. (1/2 point) How does your choice for <distance> affect item-based collaborative filtering on the MovieLens 100K data? Does this correspond to the intuition you developed in problem 2.B? (Yes…in problem 2.B you used Euclidean vs. one based on Pearson correlation, but did you developed an intuition about Manhattan vs Pearson from this?)

D. (1/2 point) How does your choice for <i> affect user-based collaborative filtering on the MovieLens 100K data? Is this result what you would have expected? Why or why not?

E. (1/2 point) Use the best settings learned from parts C and D. Varying <k> by powers of two: 1,2,4,8,16,32. What value for <k> is best for user-based collaborative filtering?

F. (1/2 point) Use the best settings learned from parts C, D, E for both item-based and user-based filters. Compare user-based to item-based collaborative filtering. Which is better? How confident are you in this result? Why or why not?