**CSC478 Final Project Proposal**

**Shan Gao, Dachi Xu, Xingyue Duan**

In this project we would like to use Movie Rating data from www.movielens.org.

The link is <http://facweb.cs.depaul.edu/mobasher/classes/ect584/data/movielens.zip>

**1. Data Set description:**

|  |  |
| --- | --- |
| u.data | The full data set, 100000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies. Users and items are numbered consecutively from 1. The data is randomly ordered. This is a tab separated list of the form: user id | item id | rating | timestamp.  The time stamps are unix seconds since 1/1/1970 UTC |
| u.info | The number of users, items, and ratings in the u data set. |
| u.item | Information about the items (movies); this is a tab separated list of the form:  movie id | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller | War | Western |  The last 19 fields are the genres, a 1 indicates the movie is of that genre, a 0 indicates it is not; movies can be in several genres at once. The movie ids are the ones used in the u.data data set. |
| u.genre | A list of the genres. |
| u.user | Demographic information about the users; this is a tab separated list of the form:  user id | age | gender | occupation | zip code  The user ids are the ones used in the u.data data set. |
| u.occupation | A list of the occupations. |
| ua.base  ua.test  ub.base  ub.test | The data sets ua.base, ua.test, ub.base, and ub.test split the u data into a training set and a test set with exactly 10 ratings per user in the test set. The sets ua.test and ub.test are disjoint. |
|  |  |

**2. Data Analysis:**

In this part we will try to analyze the data sets from different perspectives:

* Preprocess the data, preparation the data set, and explore the general characteristics of the data as a whole: examine the means, standard deviations, and other statistics associated with the numerical attributes; show the distributions of values associated with categorical attributes; etc. Use histograms or bar-graphs or cross-tabulations to plot the distribution of movie and user data.
* Perform basic correlation analysis among the movie and user attributes. Try to find out any significant positive or negative correlations among pairs of attributes. Try to construct a Correlation Matrix. Use Matplotlib library and/or ploting capabilties of Pandas, create a scatter plot of the correlated attributes.
* Experiment the movie data set with various classifiers, as well as with some of the model evaluation capabilities.
  + Run scikit-learn's KNN classifier on the test set. Generate the confusion matrix and visualize it using Matplotlib, as well as the classification report. Compute the average accuracy score. Experiment with different values of K and the weight parameter for KNN to see if we can improve accuracy.
  + Repeat the classification using scikit-learn's decision tree classifier and the naive Bayes classifier.
  + For each model, compare the average accuracry scores on the test and the training data sets. Try to figure out the bias-variance trade-off.
* Perform standard linear regression on user data. Compute the RMSE value on the full training data. Plot the correlation between the predicted and actual values of the target attribute. Display the obtained regression coefficients (weights). Finally, perform 10-fold cross-validation using the learned model and compare the final RMSE to the training RMSE.
* Perform K-means clustering on the movie training data. Display the top N terms in each cluster, percentage of movies in the cluster in which the genre appear, and the size of the cluster. Using your cluster assignments as class labels, categorize each of the documents in the movie test data into each of the appropriate cluster.
* Perform 5-fold cross-validation on the rating data set and compare MAE results using standard item-based collaborative filtering with results using the SVD-based version of the rating item-based CF. Write a new function "print\_most\_similar\_movie" which takes the movie ratings data, a query movie id, a parameter k for the number of nearest neighbors, and a similarity metric function, and prints the text of the query movie as well as the texts of the top k most similar movies based on user ratings.

**3. Application Development:**

We will try to build one applications.

* A collaborative filtering recommender system which predicts what movies a user will like or how he/she will rate a new movie based on his/her similarity to other users. A key advantage of the collaborative filtering approach is that it does not rely on machine analyzable content and therefore it is capable of accurately recommending movies without requiring an "understanding" of the movie itself.

And optionally we will also try to build a content-based recommend system:

* A content-based recommender system. Keywords are used to describe the movies and a user profile is built to indicate the type of movies this user likes. The system will try to recommend movies that are similar to those that a user liked in the past (or is examining in the present). In particular, various candidate movies are compared with items previously rated by the user and the best-matching items are recommended.