Movie Rating Predictions

Radhika Patil

(SUID: radhikap)

CS 102 Working with Data - Tools and techniques

6/1/2020

Summary

I started with figuring out what aspects of human behavior matter in rating movies. For example, some people are generally tougher than others. Some people like certain genre, others find newer movies boring and prefer older ones. Thus, a user's preferences that could matter and their overall attitude (tough vs lenient) mattered. For accounting the different features I tried neural networks, nearest neighbors, decision forests, etc. And for their attitudes, I used linear mappings, averages and cosine similarities. The best I have achieved is with linear mapping of users and movies ranges, predicting user's average rating on movie scale. It gave 90 correct predictions out of 200 in the predict set, and the fractional distance was 0.735. Ironically, accounting for features in neural networks and KNN's did not do better than plain mappings.

Data Preparation

Data was loaded in the Jupyter notebook using pandas. The NaNs in the dataset were replaced by strings. Genre and gender were converted to numbers. The movies, ratings and users dataframes were merged together. Similarly, predict, ratings and users dataframes were also merged together.

Further processing was done using either original or merged dataframes.

Techniques and Tools

Tools

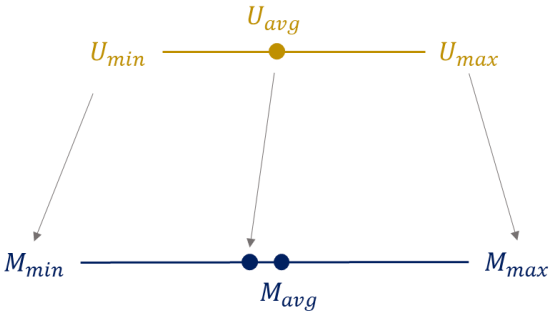
1. Jupyter notebooks for coding and processing
2. Numpy for matrix operations in cosine similarity
3. Pandas for data preprocessing, and processing
4. Scikit-learn, keras, tensorflow, for Neural Networks, KNN, Random Forest
5. Excel Spreadsheet for post processing rounding off

Techniques

I tried a couple of techniques. I will list them in the order of performance in descending order (best to worst).

Linear Mapping for user and movie ratings

The idea of this method is –



Map user rating behavior over movie rating behavior and use user average and movie average rating to generate prediction. A schematic shown on right describes the ides. U is rating for user and M is ratings for movies. A linear mapping is generated such that the input of user's minimum rating will correspond to movie's minimum and user's maximum to movie's minimum. Then the average of user is also mapped using the function and it is used as the movie's rating to predict. Essentially however, this mapping still assigns the user's average rating to the movie but accounting for how the user's average rating would map in the movie's range.

The equation is as follows

$$Prediction = (mx + c)(U_{min} \neq U_{max}) + U_{max}(U_{min} = U_{max})$$

$$m = \frac{M_{max} - M_{\min}}{U_{max} - U_{\min}}$$

$$c = \frac{U_{max}M_{min} - U_{\min}M_{max}}{U_{max} - U_{min}}$$

$$x = U_{avg}$$

I tried other variations of this strategy by mapping movie ratings to users (reverse), or predicting M_{avg} when $(U_{min} = U_{max})$, or adding standard deviation to the predictions depending on if the $M_{avg} > U_{avg}$ or $M_{avg} < U_{avg}$ etc. But the plain simple user linear mapping described above worked best.

I finally used excel spreadsheet to round of the predictions to integer values and used them for V1predict and V2predict. This gave 90 predictions correct in predict dataset with fractional distance of 0.735.

Simple user rating average and median

This strategy predicted the mean of user's average and median rating

$$Prediction = \frac{U_{avg} + U_{median}}{2}$$

Simple movie rating average and median

This strategy predicted the mean of movie's average and median rating

$$Prediction = \frac{M_{avg} + M_{median}}{2}$$

Neural Networks

Here I used a neural network with following features

Features = ['movieID', 'year','genre1_int','genre2_int','genre3_int','age','gender','userID']

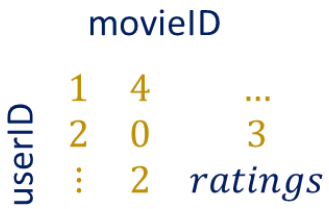
I tried a variety of parameters and the one that worked best was

num_layers = 4 , num_epochs = 30, batchsize = 10 , layer_outputs = 50

This gave me 88 predictions correct out of 200 in the predict dataset.

User Cosine similarity

I constructed a matrix with userID as row numbers and movieID as column numbers. The matrix values are the corresponding ratings. Users that had not rated a movie were assigned 0 for the value.



To predict, the row of user of the desired prediction and its dot product was taken with every other row in the matrix and cosines were calculated.

$$cosine = \frac{User.User2}{|User|_2|User2|_2}$$

The users were ranked in descending order so that the users who were closest in behavior were ranked higher. Then taking these users, predictions were made for the User.

I tried taking mean of top 10, top 50, top 100, etc. as prediction. I also tried weighting the ratings with their respective cosines, so that higher correlation of the users gave more weightage to their rating. I also tried other linear weights while taking mean. Other variations like using standard deviation along with mean did not perform as well.

K nearest neighbors classifier/regressor

The KNN classifier and regressor were used with StandardScalar transformed features. The neighbor numbers were varied for 1-500 with different features. A custom distance function was also tried, but that made the code extremely slow. The custom distance performed well on some datasets and did not on others, but turned out it almost always predicted a 4.

Decision tree and Random Forest classifier/regressor

The random forest classifier and regressor were also used with StandardScalar transformed features. Different feature values were tried.

Evaluation

I split my rated movies data in 80:20 or 95:5 and 99:1 and 99.9:0.1 ratios at different times to evaluate the models using pandas sample function. The accuracy of the models varied on parameters and the split ratio, but the ranges of evaluation were as follows.

The accuracy for neural networks KNN and decision tress was evaluated through the prediction accuracy in scikit learn. The others were evaluated by following equation.

$$Fractional\ distance = \frac{\sum|actual - prediction|}{len(test\ set)}$$

Technique	Accuracy range/fractional distances
Linear Mapping for user and movie ratings	0.76-0.78
Simple user rating average and median	0.78-0.79
Simple movie rating average and median	0.78-0.79
Neural Networks	30-37%
User Cosine similarity*	-
K nearest neighbors classifier/regressor	18-37 % or 0.86-1.12
Decision tree and Random Forest classifier/regressor	0.93 – 1.03

* The cosine similarity was difficult to evaluate with split data as the splits often contained users or movies not seen in training data and the resulting predictions would be NaNs.

Description of files used

The files used are given below. The output files for different predictions are not listed as I tried over 40 different predictions. They were generated in jupyter notebook as per requirement.

1. Predictions4.ipynb – used to code and analyze the data.
2. movies.tsv, users.csv, rating.csv, predict.csv – supplied data
3. Final files for V1predict.csv, V2predict.csv
    a. Excel spreadsheets were used to round off some of these files for integer conversion.