WHO'S BEHIND THAT MOVIE

by
Ignacio Perez
Aisha Kigongo
05/17/2013

INTRODUCTION

Movie lovers face a bewildering array of movie choices when they visit any of the systems or places available to rent or buy movies. In particular, on-demand streaming media services have a huge challenge recommending movies to new users without the users profiles. According to the literature, this *"cold start"* problem is ubiquitous in recommender systems. In this project we are interested in a subset of the *"cold start"* problem. When new movies, (which have not yet received any ratings from the community) are advertised in a catalog, they should automatically be assigned a rating. based on the ratings assigned by the community to similar movies. The similarity between movies in this instance would be determined based on the movies' content-based characteristics. In particular, we are interested in using the experience of the movie personnel (directors, screenwriters, and producers) as the input of content based predictors.

DATA

For this project, we used two datasets, Movielens and the IMDb as defined below

1. **Movielens**

   The GroupLens Research provides a number of collections of movie ratings data collected from users of MovieLens. The data provide movie ratings, movie metadata (genre and year) and demographic data about the users such as the occupation, gender, zip code, and age.

   The MovieLens data sets are divided into three smaller datasets:

   - MovieLens 100k consists of 100,000 ratings by 1000 users on 1700 movies.
   - MovieLens 10M consists of 10 million ratings applied to 10,000 movies by 72,000 users.

   We did not use the 1M MovieLens data set. All the data sets are spread across 3 tables: ratings, user information, and movie information.

2. **IMDb**

   The IMDb (Internet Movie Database) is an online database of information related to movies, television shows, actors, production crew and video games. Information is stored in a lists format that is very descriptive of the film personnel and the individual

filmographies. The lists contained included but not limited to ***the directors list, producers list, actors list*** *and so on.*

We loaded, sliced, diced and merged the data collected from the two data sets to prepare the data for our vector analysis and the bayes classifier.

METHODOLOGY

We started our work by evaluating similarity using vector analysis (Cosine distance) and then Naive bayes classifier methods to predict the rating that a movie will have based on the experience of the crew working on it.

1. **Vector Analysis**

   The primary intention of this analysis is to decompose users preferences and movies into a set of latent factors (genres), and verify if it is possible to correlate the rating that the user gave to a movie with the movie itself. We are expecting something really simple: A shorter distance between the vector of the user and the vector of the movie should imply a better rating.

   Vector analysis required pre-processing of the ratings matrix before before it could be used in a matrix function. The preprocessing functions collected data summary statistics across rows and columns of the matrix.

   In our analysis, we created vectors of experience per film personnel per movie and used the average rating for the weights of each movie in a specific genre. We tallied up the weights per genre and then normalized the weighted experience vector by dividing it by the count of the most popular genre keeping the rating of the that genre constant.

   The figure below shows an example of 2 directors and their weighted experience vectors. We use the same method to calculate the vector for each movie. We will only consider the experience of the crew previous to the year of release of the movie

		Action	Mystery	Thriller	Crime	Comedy	Drama	Romance	Rating
Tom Hanks	Angel & demons		1	1					**6.6**
	Joe v. the Volcano					1		1	**5.5**
	Forrest Gump						1	1	**8.7**
	Average Experience Vector		**6.60**	**6.60**		**5.50**	**8.70**	**7.10**	
	Weighted Experience Vector		**3.30**	**3.30**		**2.75**	**4.35**	**7.10**	
Liam Neeson	Taken	1		1	1				**6.2**
	Chloe		1	1			1		**6.3**
	The A-team	1		1					**6.8**
	Average Experience Vector	**6.50**	**6.30**	**6.43**	**6.20**		**6.30**		
	Weighted Experience Vector	**4.33**	**2.10**	**6.43**	**2.07**		**2.10**		
Movie	**X**	**2.17**	**2.70**	**4.87**	**1.03**	**1.38**	**3.23**	**3.05**	
	Movie intention		**2.70**	**4.87**					

Figure 1

		Action	Mystery	Thriller	Crime	Comedy	Drama	Romance	Rating
Movie	**X**	**2.17**	**2.70**	**4.87**	**1.03**	**1.38**	**3.23**	**3.50**	
	Movie intention		**2.70**	**4.87**					
User	Angel & demons			1					**6**
	Zoolander		1			1		1	**7**
	Braveheart						1	1	**10**
	User Vector		**7.0**	**6.0**		**7.0**	**10.0**	**8.5**	
	Weighted User Vector		**3.5**	**3.0**		**3.5**	**5.0**	**8.5**	

Figure 2: User Vector

We are interested in the the relation between the Cosine distance and the ratings. To get the cosine distance we measured the distance between two vectors (user vector and the movie vector) of an inner product space that measures the cosine of the angle between them. We imported the scipy.spatial.distance library that defines the cosine distance as where u.v is the dot product between the two vectors

$$1 - \frac{u \cdot v}{||u||_2||v||_2}.$$

u = user vector , and v = movie vector

The cosine of 0° is 1, and it is less than 1 for any other angle. Two vectors with the same orientation have a Cosine similarity of 1, two vectors at 90° have a similarity of 0, and two vectors diametrically opposed have a similarity of -1, independent of their magnitude.

Results:

Initial attempts with vectors gave us confusing results but after the review of the data we got positive insights. We realized that:

- Introduction of actors without any filter would introduce irrelevant information into the model.
- Intuitively, the work of directors, writers and producers is 100% involved in the movie unlike actors and actresses, who have a different participation based on the importance of the character.
- The relation between cosine similarity and the ratings was less strong than the expected, even if the rating (figure 3 & 4)

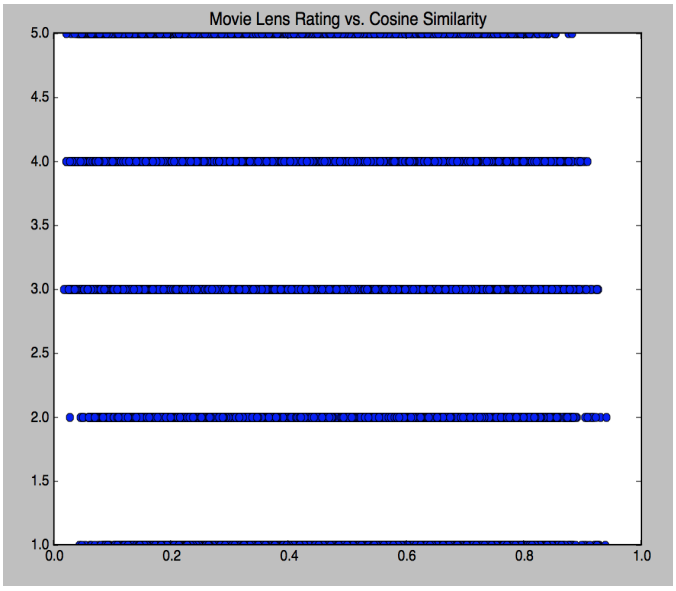

Figure 3

Note: Vertical axis => ratings; Horizontal axis => ratings

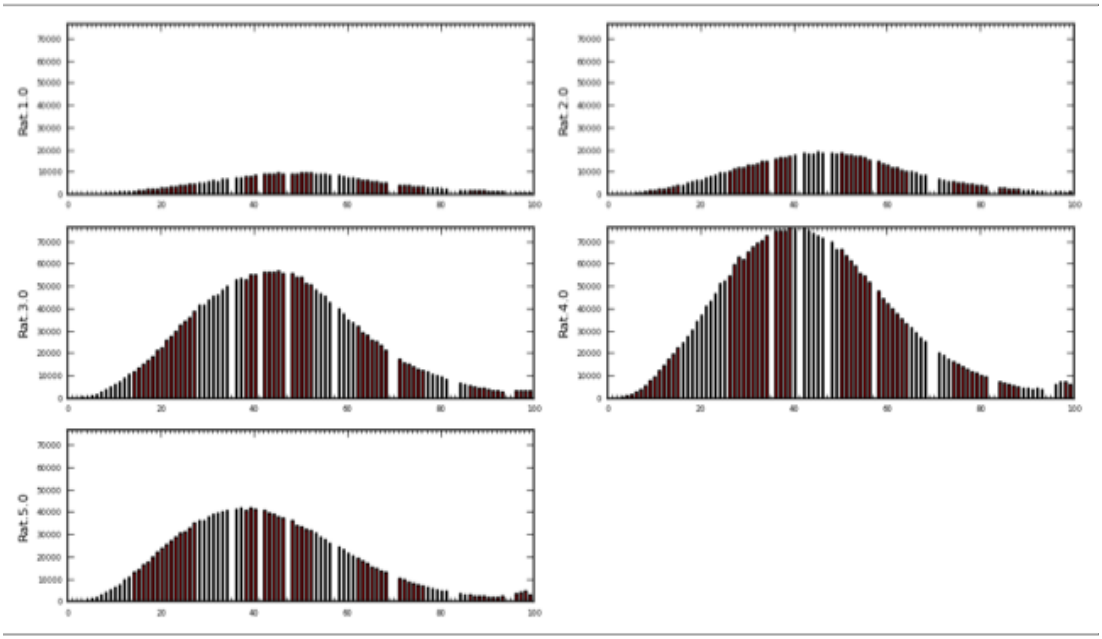

Figure 4

Note: Vertical axis => Count of reviews; Horizontal axis => cosine similarity

However the average distance vs. the rating shows a really small relation between those variables:

Rating 1.0 avg. similarity 0.50202138569
Rating 2.0 avg. similarity 0.473019643486
Rating 3.0 avg. similarity 0.45488130407
Rating 4.0 avg. similarity 0.438172432737
Rating 5.0 avg. similarity 0.425659307968

2. **Naive Bayes Classifier**

   Bayes classifier is a statistical classifier that predicts class membership probabilities such as the probability that a given tuple belongs to a particular class. It is a robust classifier with respect to missing feature values, which make it well suited to the task of rating prediction.

   To apply the naive Bayes classifier to rating prediction, we independently teach one classifier for each user u. We train the classifier for u using all movies rated by u,

excepting one target movie m. Then we predict the rating of m using the rests of the movies.

To learn the naive Bayes rating predictor we estimate P(Director is in Movie) and P(rating=1 | Director is in Movie). We smooth the probabilities by adding prior counts to avoid zero probabilities.

$$P(rating = 1|DinMovie) = \frac{P(DinMovie|rating = 1) * P(rating = 1)}{P(DinMovie|rating = 1) * P(rating = 1) + P(DinMovie|rating! = 1) * P(rating! = 1)}$$

Results

100K ratings.
10930 ratings not considered for not matching (IMDb & Movielens)

Bayes OK 21919
Bayes NoOK 67691
Total 89610
Number of Users 943
Average prediction success per user: 22%

FUTURE STEPS

For our next steps we plan to use the data to explore around. But first we'd look into improving the vectors by

1. Consider the actors/actresses dataset and evaluate an individual's amount of work on each movie (leading role v. supporting role or just a cameo to the show). We plan to extract data from wikipedia to get the plot of the movie and then compare how many times the actors role appears within the description to get the amount of work. Our assumption is if the actor's character appears often in the plot, then that is an indication of how important his role is to the movie.
2. We can add new parameters for the classifier using the cosine similarity.
3. Establish baselines for the ratings and compare them with the similarities