Who's behind that movie?

Ignacio Perez, Aisha Kigongo

Director: **Tim Burton**
Genre:

- Action
- Fantasy

Director: **Joel Schumacher**
Genre:

- Action,
- Adventure
- Crime

Director: **Christopher Nolan**
Genre:

- Action
- Thriller
- Crime

Goal

What is the predicted rating of a new movie given the experience of the crew working on it?

Crew:

- Directors
- Actors
- Screenwriters
- Producers

Data

- 2 datasets : Movielens & IMDb
    - Movielens
        - Dataset 1: 100K ratings
        - Dataset 2: 10M ratings
    - IMDb
        - Movies
        - Crew
            - Writers
            - Director
            - Producers
            - Actors
            - ...etc

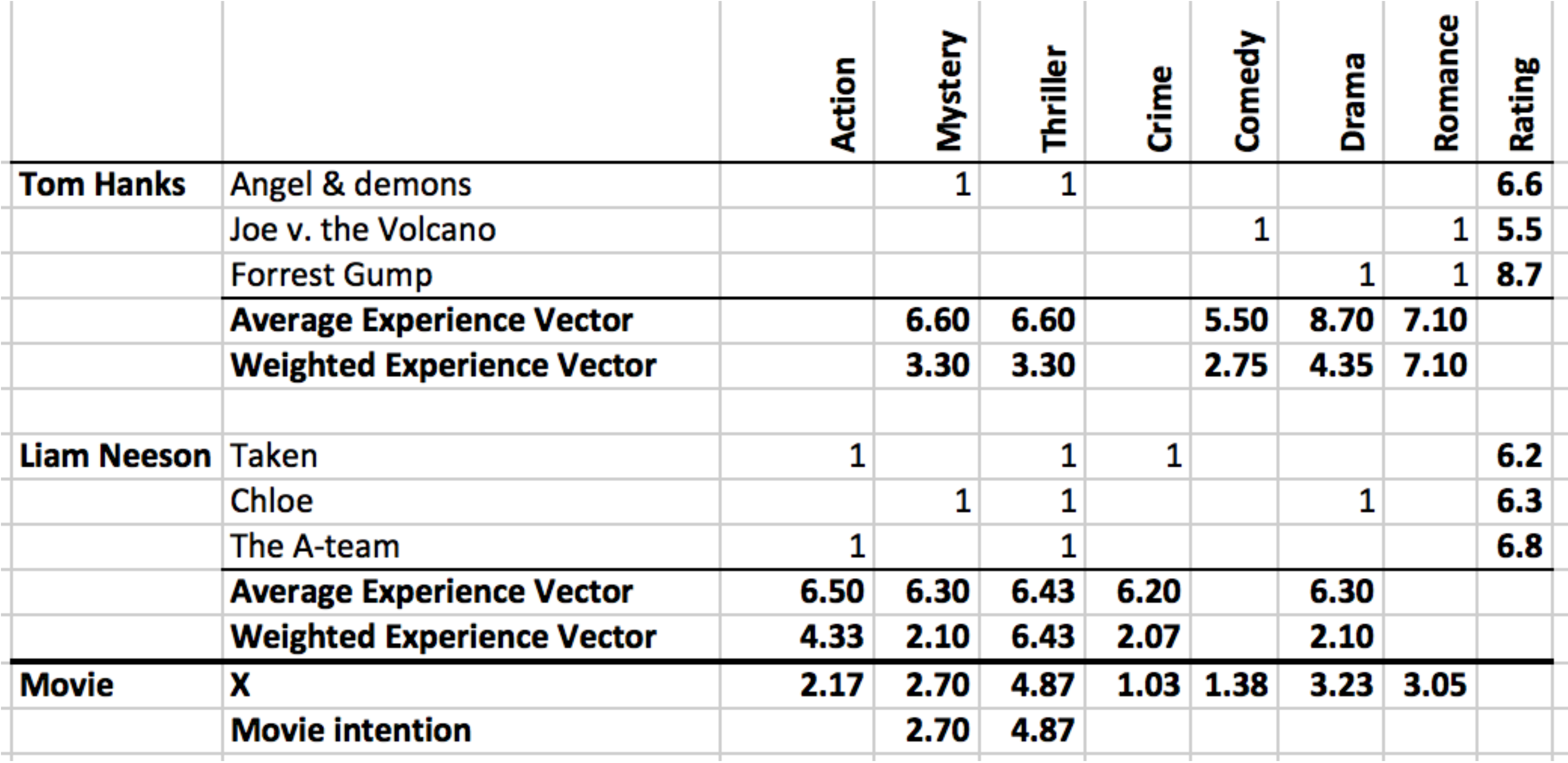
Data : Challenges

- Difficulty in parsing and loading the data.
  - IMDb data organized into very small chunks
    - *e.g directors.list, actors.list, producers.list etc*
- Different naming standards (movies, genres)
  - for example
    - *The Magnificent Seven vs Magnificent Seven, The*
    - *Seven vs Se7ven*
- Size: 4gb,
  - IMDb lists approx. 1gb
    - prepared for Map-Reduce jobs
    - organized in python arrays

Methods

- Vector Analysis
    - Cosine distance vs. Rating
- Naive Bayes Classifier
    - For the prediction

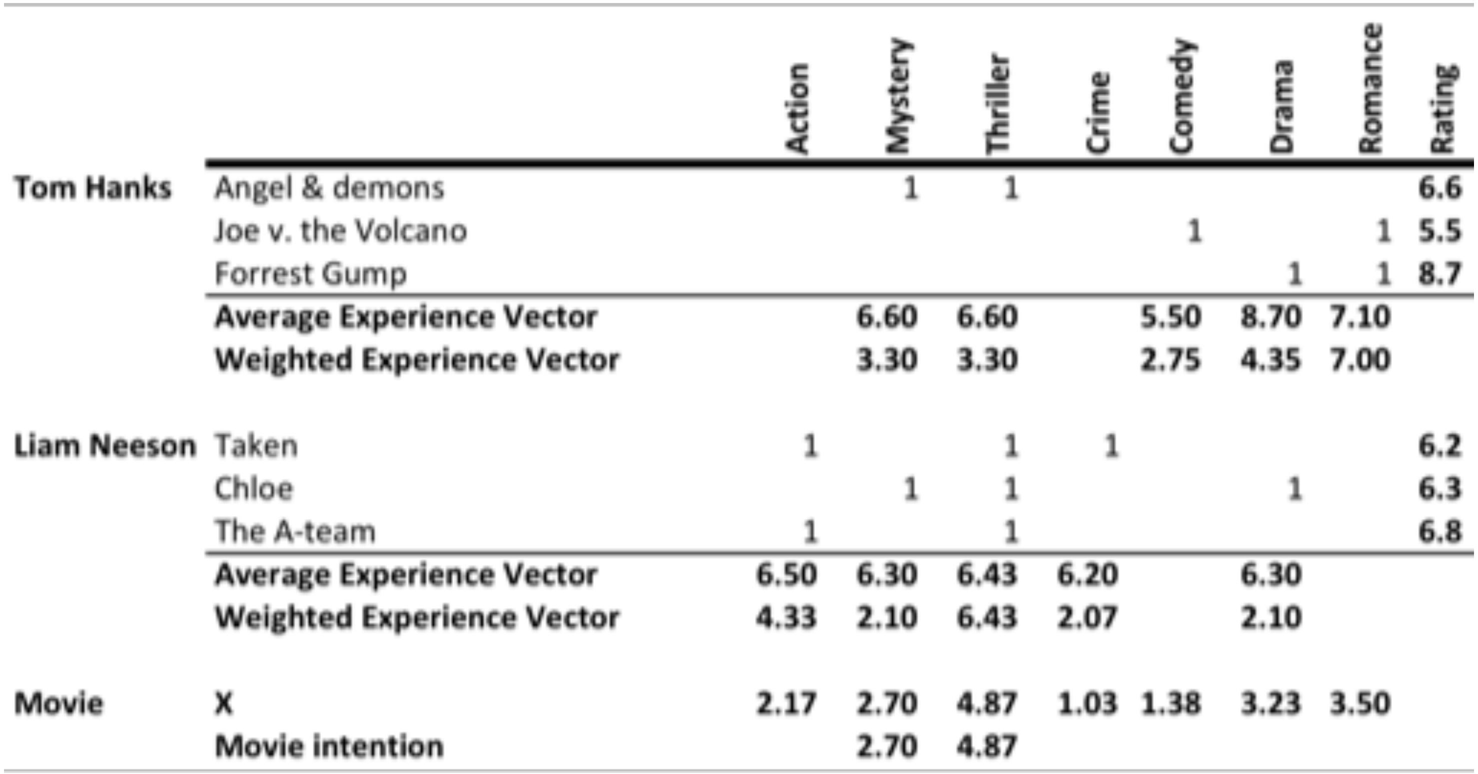
Experience Vector



		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**					

Experience

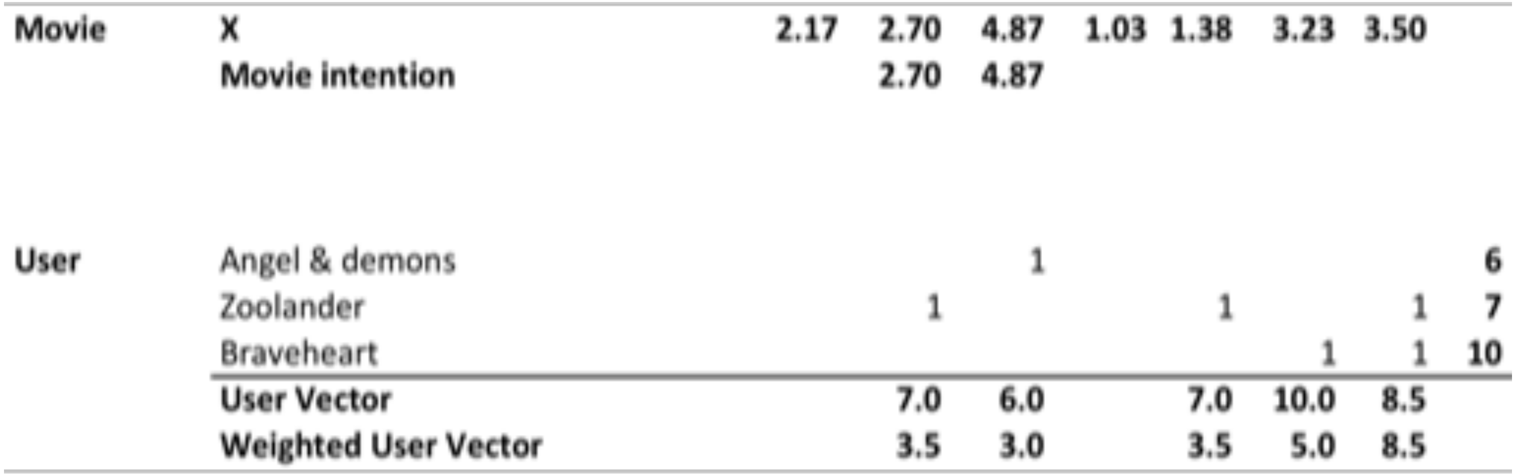
- **Dimensions** : *Genre*
- **Factor** : *average ratings*



		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**	**7.00**	
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**	**3.50**	
	Movie intention		**2.70**	**4.87**					

User Comparison

- Cosine Distance (or 1 - Cosine Similarity)
- User Weighted Vector

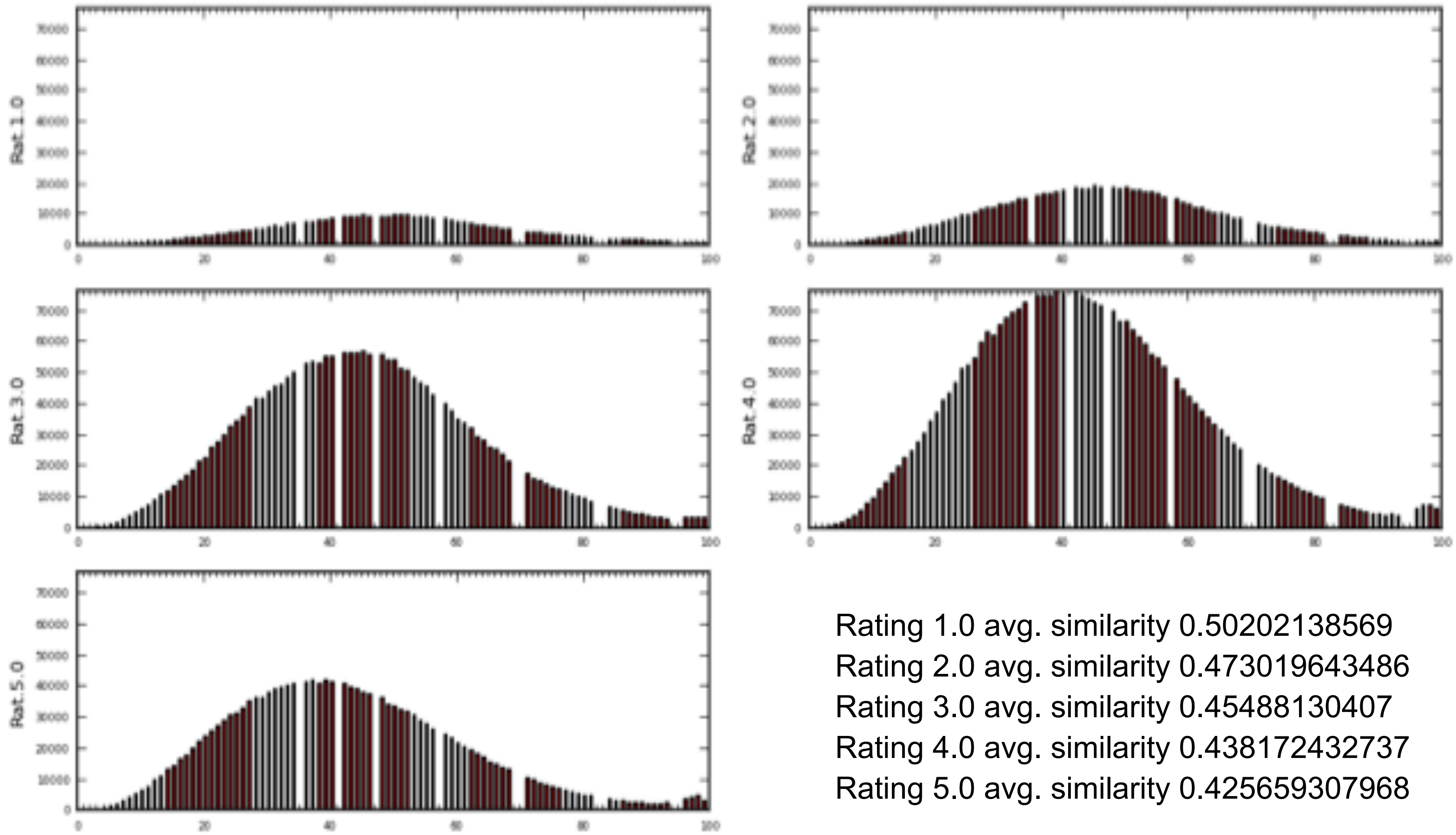


Movie	**X**	**2.17**	**2.70**	**4.87**	**1.03**	**1.38**	**3.23**	**3.50**	
	Movie intention		**2.70**	**4.87**					
User	Angel & demons			1					**6**
	Zoolander		1			1		1	**7**
	Braveheart						1	1	**10**
	User Vector		**7.0**	**6.0**		**7.0**	**10.0**	**8.5**	
	Weighted User Vector		**3.5**	**3.0**		**3.5**	**5.0**	**8.5**	

Process

- Initially the vector gave us confusing results, but after reviewing of the data, we got positive insights:
  - Inclusion of actors without filters introduced irrelevant information into the model
  - Filtering the dataset: The directors, writers and producers are 100% involved in the movie.
- The relation between cosine similarity and the ratings was less strong than the expected.
  - But allowed us to improve the next step

Cosine Distance (X) v Freq. per Rating(Y)



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

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

Classifier

- Naives Bayes Classifier
  - One classifier per user
  - We assigned a specific rating instead of working with ranges as defined in the literature

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

Methodology

For each user:

- Remove one movie from the user reviews.
    - With the rest of the crew, get the params for the classifier (individual classifier)
    - The classifier will return a probability for each rating (1-5) for the removal movie
    - The rating with the maximum probability is the answer of the classifier
- Using the movielens 100K dataset
    - **Bayes OK** 21919, **Bayes Error** 67691
    - Total 89610
    - Success 24%

Future steps

- Improve the vectors:
    - Consider the actors/actresses dataset, but evaluating the amount of "work" for each move based on the role. i.e (leading role v. supporting role)
    - New parameters for the classifier using the cosine similarity
    - Baseline rating for the vector analysis