Movie Recommendation with DBpedia

Mirizzi et al. (2012)

Students:
Lucas van Berkel
Ruben Woortmann

May 21, 2019

Supervisors:
Ilaria Tiddi
Albert Meroño Peñuela

Problem

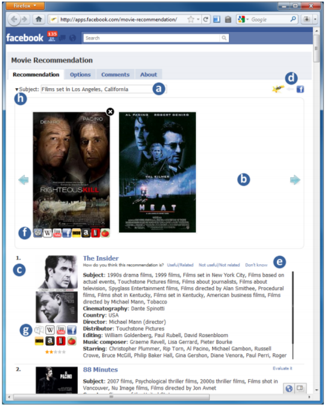
- Movie Recommender System using Linked Data
  $\rightarrow$ Tailored to users
- Content-based recommender systems face cold-start problems

Research

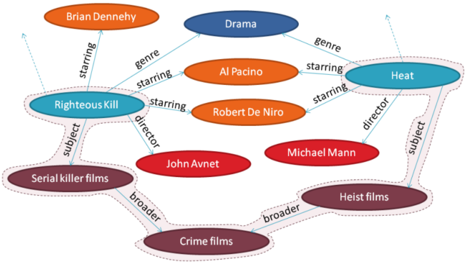
- Using a Vector Space Model to calculate similarities between movies
- Predicting movies of interest by combining movie similarities and user preferences

APPLICATION

- Users can select movie characteristics they are interested in $\rightarrow$ Genre, actors, directors
- Clicking on a movie results in similar movies
- User can indicate they like a movie
- Future recommendations will be based on liked movies

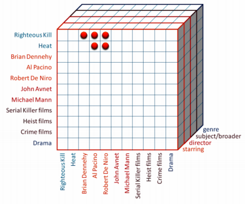


RDF-graph



- Similarity between movies is based on the amount of characteristics they share

Vector Space Model



- Each slice represents a property
- Object are rows, subjects are columns
  →Cell indicates whether object and subject are related

VECTOR SPACE MODEL

$$\overrightarrow{m_{i,p}} = (w_{1,i,p}, w_{2,i,p}, \ldots, w_{t,i,p})$$

- E.g. property "starring" contains vector per movie with Boolean for all properties in DB
- Movie represented by matrix of size (#properties * #possible attributes per property)
- Similarity based on cosine of angle can be calculated per property or overall

USER PREFERENCES

$$\tilde{r}(u, m_i) = \frac{\sum_{m_j \in profile(u)} \frac{1}{P} \sum_{p} \alpha_p \cdot sim^p(m_j, m_i)}{|profile(u)|}$$

- Reweighting the similarity given the user preferences

Weight Training

$$w_{n,i,p} = f_{n,i,p} * \log\left(\frac{M}{a_{n,p}}\right)$$

- Weights found through genetic algorithm and 5-fold cross-validation
- Weights applied to movie characteristics based on their rarity
- Weights can be adjusted by filtering for certain properties

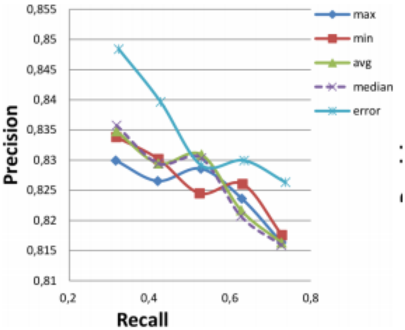
CROSS-VALIDATION

	$\alpha_{subject}$	$\alpha_{director}$	α_{writer}	$\alpha_{starring}$	$error$
α^1	0.123	0.039	0.080	0.159	3
α^2	0.024	0.061	0.274	0.433	5
α^3	0.267	0.356	0.188	0.099	3
α^4	0.494	0.428	0.244	0.230	4
α^5	0.082	0.457	0.484	0.051	1

Table 1. Example of values computed after the training.

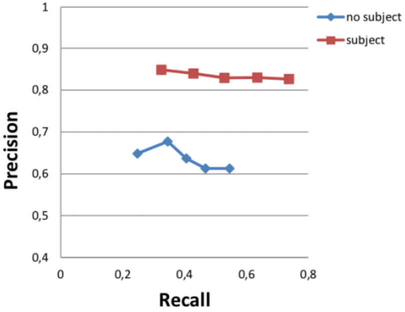
- Movielens dataset split in five folds
- Trained weights for every property

Assessment



- MovieLens contains per user ratings of movies
- Multiple top-N recommendations were assessed through precision and recall

Conclusion



- The algorithm performed considerably better with ontology information enabled
- System should be able to generalise as a recommender system for any linked dataset