Automatic movie ratings prediction using machine learning

Mladen Marović, Marko Mihoković, Mladen Mikša, Siniša Pribil, and Alan Tus
University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, Zagreb, Croatia
Email: {mladen.marovic, marko.mihokovic, mladen.miksa, sinisa.pribil, alan.tus}@fer.hr

***Abstract*—Recommendation systems that model users and their interests are often used for the improvement of various user services. Such systems are usually based on the automatic prediction of user's ratings of the items provided by the service. This work presents an overview of some of the methods for automatic prediction of ratings in the domain of movie ratings. Chosen methods are based on various approaches described in related work. While working, the methods use both users and the items features. For the purpose of this work, data was gathered from the publicly available movie database IMDb. Within is an implementation of the chosen methods and their evaluation using the gathered data. The results show an improvement in comparison to the chosen baseline methods.**

I. Introduction

II. Predicting movie ratings

Li and Yamada created a system to predict movie ratings based on decision trees [**?**]. Decision trees use selected film features when predicting a movie rating. User preferences for the above mentioned film features were modeled by building a separate decision tree for each user.

For their hybrid system, [**?**] are proposing a model which contains user and movie features, as well as all available ratings. Rating prediction is then done by using *singular value decomposition method* (*SVD*) and *k nearest neighbors algorithm* (*kNN*). Authors call this method a *SVD-kNN* method.

III. Methods for movie ratings prediction

A. Content based methods

One of the basic content based methods used in this work is the regression tree. Each tree represents a user profile created using the users ratings. The features used in this work are: genres, actors, directors and scenarists. All features are represented using a binary vector, meaning that each row corresponds to a genre, actor, director or scenarist. The rows of each vector that fit a movie are marked with a one (1) and the rest is marked with a zero (0). When building a regression tree the minimum square error criterion was used. Classification trees were taken into consideration as well but preliminary testing showed that they produced inferior results.

B. Collaborative methods

Personality diagnosis method, taken from [**?**], models user's personality and uses it in the prediction of ratings. If there are m movies in total, personality type of user i can be presented as a vector $\mathbf{R}_i^{\mathrm{true}} = \langle R_{i1}^{\mathrm{true}}, R_{i2}^{\mathrm{true}}, \ldots, R_{im}^{\mathrm{true}} \rangle$, where the ratings R_{ij}^{true} represent "true" ratings of the movie j for the user i. Additionally, it is assumed that the actual user ratings have Gaussian noise added to the "true" ratings. For user i and movie j, the recorded rating presents a realization of the random variable, governed by normal distribution, whose mean value is equal to the "true" rating R_{ij}^{true}:

$$P(R_{ij} = r | R_{ij}^{\mathrm{true}} = r_t) \propto e^{\frac{-(r-r_t)^2}{2\sigma^2}}.$$

It is assumed that the acquired rating vectors form a representative distribution of personality types present in the population. User u for which the prediction is being made can belong to any of the observed personalities $\mathbf{R}_1, \ldots, \mathbf{R}_n$ with probability $1/n$. Knowing some of the ratings of user u, the probability that he belongs to personality type $\mathbf{R}_i$ can be calculated by:

$$\begin{aligned} &P(\mathbf{R}_u^{\mathrm{true}} = \mathbf{R}_i | R_{u1} = r_1, \ldots, R_{um} = r_m) \\ &\propto P(R_{u1} = r_1 | R_{u1}^{\mathrm{true}} = R_{i1}) \\ &\qquad \cdots P(R_{um} = r_m | R_{um}^{\mathrm{true}} = R_{im}) P(\mathbf{R}_u^{\mathrm{true}} = \mathbf{R}_i). \end{aligned}$$

Finally, the probability distribution of the rating for movie i and current user u can be estimated using previous probabilities for personality types as follows:

$$\begin{aligned} &P(R_{ui} = r_i | R_{u1} = r_1, \ldots, R_{um} = r_m) \\ &= \sum_{j=1}^{n} P(R_{ui} = r_i | \mathbf{R}_u^{\mathrm{true}} = \mathbf{R}_j) \\ &\qquad \cdot P(\mathbf{R}_u^{\mathrm{true}} = \mathbf{R}_j | R_{u1} = r_1, \ldots, R_{um} = r_m). \quad (1) \end{aligned}$$

The rating with highest probability is then the predicted rating of user u for movie i. The method is defined only with the deviation of normal distribution σ.

Probabilistic latent semantic analysis was implemented according to [**?**]. Intuitively, latent variables represent hidden causes for ratings that user gave to each movie. Users are divided into groups according to similar rating causes using latent variables. Formally, probability that a user u rates movie i with rating r can be written as:

$$p(r|u,i) = \sum_z p(r|i,z) P(z|u),$$

where z is latent variable which represents hidden causes. Probability that the cause z is responsible for the ratings of user u is presented by term $P(z|u)$, while $p(r|i,z)$ is a probability distribution for the ratings of a known movie i according to the cause z. It is assumed that this probability distribution is a normal distribution. Expectation maximization algorithm is used for the estimation of distribution's parameters $\mu_{i,z}$, $\sigma_{i,z}$ and probability $P(z|u)$. E-step of the algorithm is represented by equation:

$$P(z|u,r,i;\hat{\theta}) = \frac{\hat{p}(r|i,z)\hat{P}(z|u)}{\sum_{z'} \hat{p}(r|i,z')\hat{P}(z'|u)},$$

where the hat symbol presents probabilities according to parameters $\hat{\theta}$, that is, previous estimations for the probability's parameters. Probabilities of latent variables z for all recorded ratings are calculated using the stated equation. This probabilities are then used in the M-step for the estimation of probabilities $P(z|u)$:

$$P(z|u) = \frac{\sum_{\langle u',r,i\rangle : u'=u} P(z|u,r,i;\hat{\theta})}{\sum_{z'} \sum_{\langle u',r,i\rangle : u'=u} P(z'|u,r,i;\hat{\theta})}$$

and distribution parameters $\mu_{i,z}$ and $\sigma_{i,z}$:

$$\mu_{i,z} = \frac{\sum_{\langle u,r,i'\rangle : i'=i} r\ P(z|u,r,i;\hat{\theta})}{\sum_{\langle u,r,i'\rangle : i'=i} P(z|u,r,i;\hat{\theta})},$$

$$\sigma_{i,z} = \frac{\sum_{\langle u,r,i'\rangle : i'=i} (r-\mu_{i,z})^2\ P(z|u,r,i;\hat{\theta})}{\sum_{\langle u,r,i'\rangle : i'=i} P(z|u,r,i;\hat{\theta})},$$

where $\langle u,r,i\rangle$ represents the set of all recorded ratings. All ratings are normalized according to the mean rating μ_u and deviation σ_u of the user according to equation:

$$(u,r,i) \mapsto (u,r',i), \quad \text{using} \quad r' = \frac{r-\mu_u}{\sigma_u}. \tag{2}$$

Using the calculated parameters, the predicted rating can be obtained as a mathematical expectation for $p(r|u,i)$:

$$E[r|u,i] = \mu_u + \sigma_u \sum_z P(z|u)\mu_{i,z}.$$

C. Hybrid methods

Hybrid method *SVD-kNN*, described in [**?**], was implemented for the purposes of this study. All available user and movie features are placed in the matrix:

$$H_{m\times n} = \begin{bmatrix} R_{u\times i} & U_{u\times y}\cdot w_2 \\ I_{x\times i}\cdot w_1 & 0_{x\times y} \end{bmatrix}, \begin{pmatrix} m = u+x \\ n = i+y \end{pmatrix}$$

where $R_{u\times i}$ is a matrix of ratings which users gave to movies they watched, $U_{u\times x}$ is a user feature matrix, $I_{x\times i}$ is a movie feature matrix and w_1 and w_2 are weighted factors that are used to define at which ratio do movie and user features affect on prediction result. Note that filling matrix H with zeros $0_{x\times y}$ is necessary to preserve its rectangular shape. Figure 1 shows an example of matrix H.

In order to achieve better rating predictions, it is necessary to remove user preferences, movie popularities and other global effects from matrices R, U and I. For rating matrix R this is done by rating values in form of subtracting weighted combination of overall- ($\bar{r}$), user- ($\bar{r}_u$) and movie-average ($\bar{r}_i$) from the original rating:

Fig. 1. Primjer matrice na kojoj se temelji *SVD-kNN*.

$$\tilde{r}_{ui} = r_{ui} - \alpha\bar{r} - \beta\bar{r}_u - \gamma\bar{r}_i.$$

The parameters α, β and γ determine the influence of each effect. Feature matrices, U and I, consist only of values 0 or 1, so previously mentioned normalization method is not applicable here. Instead, the following expression is used:

$$\tilde{F} = MFN,$$

where F is a user- or movie-feature matrix and M and N are diagonal matrices with values:

$$M_{xx} = \frac{1}{\sqrt{\sum\limits_n F_{xn}}} \quad \text{i} \quad N_{yy} = \frac{1}{\sqrt{\sum\limits_m F_{my}}}.$$

After removing global effects in rating and feature data, *SVD* method is used on resulting matrix H. *SVD* is an eigenvalue generalization for non-square matrices which is often used in signal processing and statistics. Linear decomposition procedure factorizes starting matrix into three low-dimensional resulting matrices containing left-singular vectors (V), singular values (S) and right-singular vectors (W) respectively:

$$H_k = V_{m\times k}\cdot S_{k\times k}\cdot W^T_{k\times n}.$$

Parameter k defines number of extracted eigenvectors and is crucial for model accuracy. Matrices V, S and W are used to calculate compound matrices $X = [VS^{\frac{1}{2}}]$ and $Y = [S^{\frac{1}{2}}W^T]$ which can be considered as separate user and movie concepts and use as a basis for collaborative rating prediction methods. Using the k nearest neighbor algorithm, prediction is done with expression:

$$\hat{r}_{ui} = \frac{\sum\limits_{j\subset ocijenjeniFilmovi(u)} s_{ij}r_{uj}}{\sum\limits_{j\in ocijenjeniFilmovi(u)} |s_{ij}|},$$

where s_{ij} is a cosine similarity defined analogous to expression (**??**). Model accuracy is also dependent on the number of neighbor elements (n) that are included in the calculation.

IV. Evaluation

All methods mentioned before are implemented and tested using a *Matlab* software package. This section describes training and test data, as well as performance measures that are used in experiments. Last subsection contains experiment results and their comments.

A. *Training and test data*

Movie and user data is collected from publicly available IMDb[1]. Movie ratings in collected data are represented by the ratings that users gave in their reviews. Apart from the ratings, following movie features are collected: title, genres, year of release, directors, scenarists and actors. It is possible that some movies do not have all of the listed features. Although bigger data set is collected, the set of 1059 users, 9428 movies, 1000 actors, 1066 directors, 1060 scenarists, 28 genres and 65581 ratings in range from 1 to 10 is used for the purposes of this work.

Training data set is generated using *AllButOne* method described In One rating from every user in training set is left out and used to create a test set. Thus the training set contains 64522 ratings, while test set contains 1059. Average number of ratings per movie is 7.57, per user is 60.93, and average rating value is 7.22.

B. *Performance measures*

C. *Results*

The regression tree was trained using ten folded cross-validation that showed the best places to trim the tree. The results obtained in this work are inferior to baseline methods. Such a result can be explained with too few training samples per user and a very sparse distribution of features per film. A small number of sparse vectors cause problems when discerning a movie.

V. Conclusion

Predicting user ratings for some object presents an interesting and well formed problem. Many different approaches to solving this problem were used with differences in used methods and given results. This works gives a comparison of a number of methods used in automatic prediction of movie ratings. The used dataset was collected from the publicly available IMDb. While testing, the probabilistic latent semantic analysis proved as best.

Future plans include more detailed testing with different datasets because the currently used dataset does not contain a required diversity in user ratings. Some of the implemented methods were not tested thoroughly enough because of their computational complexity and should be tested in detail. Other methods suggested in related work should be taken into consideration as well. Finally, a combination of classifiers could be tested and the use of weighed voting when predicting.

[1]`http://www.imdb.com/`