EEE - 485

STATISTICAL LEARNING AND DATA ANALYTICS

PHASE I REPORT

KEREM AYÖZ - 21501569
MÜCAHİT FURKAN YILDIZ - 21400425

Retro Movie Recommender

1. Problem Description and Dataset

In retro movie recommender Project, we will try to implement a learning program to recommend movies to every user of our program. Watching movie is a very common and enjoyable event for idle times with multiple participant. Therefore choosing movie is very important and hard mission. There are millions of films with different features such as categories, length, actors, tags. Choosing the right film among all them can be take a lot of time. A system that could give reasonable recommendation would be very helpful to people. As a movie lover, we thought that implementing such a program is reachable aim for us due to our knowledge on the topic. There are a lot of different ways to follow, however, we will use the way which described in the next section. It uses some methods but the combination and constructing the whole model is founded by us.

In this project, one of the important element is dataset. We searched for dataset and founded different sets with different features. We chose MovieLens datasets because of the enormous information that they contains such as millions of ratings, thousands of users and movies. However we had to change it. We found another dataset from kaggle.com[1] which is also uses MovieLens movies and additionally it has the features of movies such as posters, backdrops, budget, revenue, release dates, languages, production countries companies, and movie plot keywords in .cvs files which are usable easily. We will use these features in our project and they have significant roles.

2. Review of the Methods

We will use 3 different statistical learning methods for our project. Principal Component Analysis (PCA), Ridge Regression and Neural Network are the methods that we will use.

2.1. Collaborative Filtering

We will extract a matrix which contains movies in the columns and users in the rows. Values will represent the ratings of the users to the movies. However, these matrix will be sparse since a user did not rate all the movies in our dataset. In order to fill that matrix, we will use item-based collaborative filtering.

We will first calculate the similarity matrix of items, in our case items are movies. We will use Cosine Similarity to calculate the similarity. Each movie has some features therefore they can be used as inputs for calculation of similarities between them. After calculations, we will obtain mxm matrix where m is the number of movies in our dataset, which represents the similarities between movies. Each value dij represents the similarity between movie i and movie j.

After obtaining the similarity matrix, we will use the following formula to fill the missing ratings in the user-movie rating matrix.

$$\hat{r}_{pi} = \frac{\sum_{j \in Q_i} d_{ij} * r_{pj}}{F}$$

In that formular, where (dij gelcek) expresses the similarity between the movie *i* and movie *j*. (rpj) expresses the rating that is given by the user *p* to the movie *j*. F is a normalizing factor. Therefore we calculated the missing ratings as the computing the sum of the ratings given by the user p on the movies similar to i.

2.2. User Feature Extraction

In our dataset, there are no features about the users in the dataset. In order to create a profile of a user, we will use Principal Component Analysis method. By using that method, we will obtain fixed size user-feature vector. The dimension of the vector will be fixed number, which we can take it as a hyperparameter. We will generate a feature vector for each user.

In order to generate a feature vector, we transform to the data into another coordinate systems with greatest variance is occur on the first coordinate, second most variance occurs at second coordinate. Variance of the projections is given below.

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{u})^2 &= \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{u})^T(\boldsymbol{x}_i^T\boldsymbol{u}) \\
&= \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{u}^T\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{u} \\
&= \boldsymbol{u}^T\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i\boldsymbol{x}_i^T\right)\boldsymbol{u}
\end{aligned}$$

If we maximizes this equating we get the eigenvectors of Σ as principal components. Then use them to feed neural networks.

2.3. Ridge Regression to Calculate Feature Weights

The last step before we get into the rating estimation, we will estimate the importance of the features in the movie-feature. We have various features for each movie such as budget, genres, language and more other features. However; not all the features are equally important to determine the rating of the movie. In order to differentiate the importance of the features, we will put weights to each feature and will use a Ridge Regression to learn these weights. The label of the data will be the average rating of the movie. We will use the closed form solution of Ridge Regression which is as following;

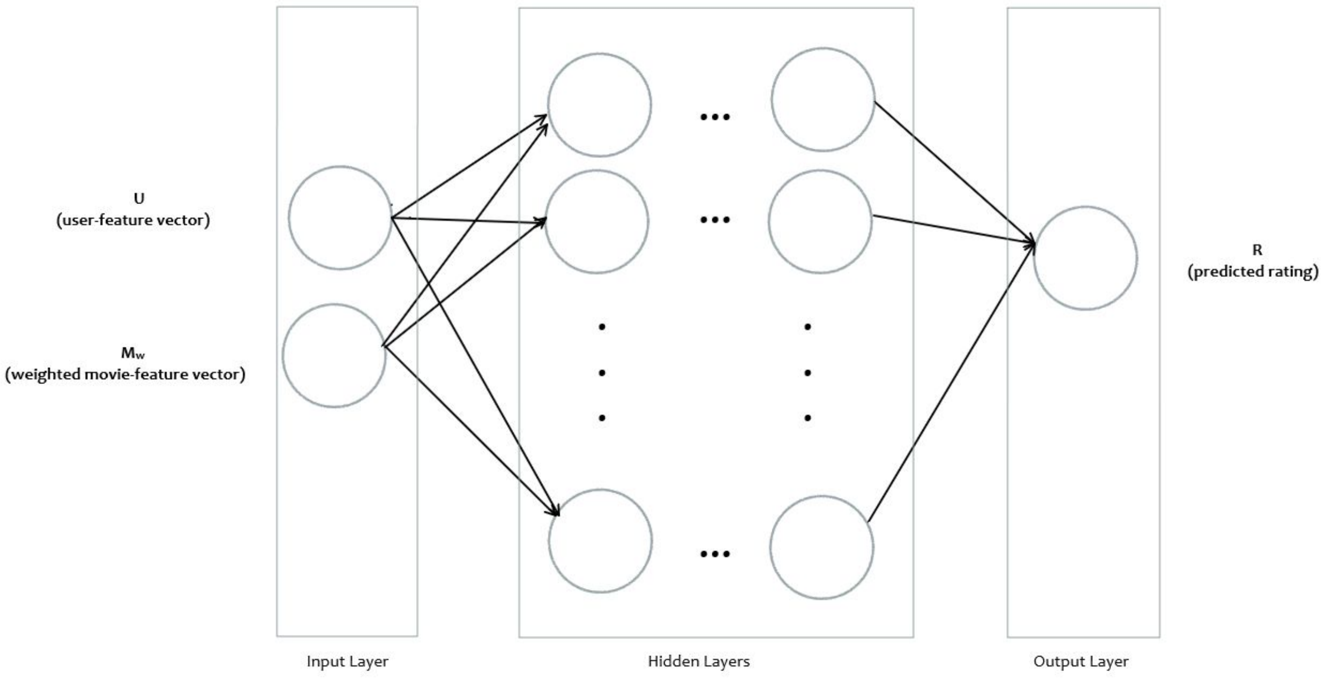
$$\hat{\beta}^R = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\boldsymbol{y}$$

where rows of the X matrix represent the feature vector of each movie and y represents the average ratings of each movie. The hyperparameter ƛ will be chosen at the test and validation stage of the project.

2.4. Rating Estimation With Neural Networks

After getting the user-feature vectors and weighted movie-feature vectors, we will train a model which takes these vectors as inputs. The label of that input data will be the actual rating that user gives to the movie. We will try to estimate the rating of the movie that the user gives.

We will have 2 neurons in the input layer and 1 neuron in the output layer. We will decide the number of hidden layers, number of neurons in that layers at the test and validation stage to choose the optimal numbers. A simple design of our model is as following;



After training the model, we now have a model that predicts the rating that user would give to a movie. Then we will take the movies with top 10 highest estimated rating and recommend these movies to the user. In training we will calculate the mean-squared error with that estimated value and actual rating that the user gives to that movie in training dataset. Also we will use the part of the dataset for testing.

3. Challenges

There are many challenges we can think and there are many that we will understand when we are on the way. At the first sight there are a lot of hyper-parameters that should be considered. We uses collaborative filter to fill the movies-users matrix and our parameter selections of F directly affects the filling and due to that directly affects the other parts of our design. These are very dangerous areas to work and we need to be very careful while working on it.

Another challenge is filling it without any feature insertion while protect the existing features. In the second part we uses principal component analysis to extract the features of the users. In this part we need to decide how many principal component we are going to use. If we uses many, it will be less effective.

In the third part, the ridge regression, the is no as much as challenges compare to the other parts. We need to decide λ only. However it can be also difficult, however we can find the convenient selection by considering the error - λ relation.

In the end we have a neural network, where a lot of hyper-parameters show up. We need to decide hidden layer numbers and neuron numbers of each layers. Implementing them will be very difficult. It may be take a long time. Another drawback of the using a neural network is data size. Out dataset has enormous information to process. If we uses all the dataset, training the algorithm can be take a long time. We have to cut the dataset to solve this problem.

4. Validation of Methods

We will use the k-fold cross validation method to validate our model. The number k that we will use in our project will be 5. This number makes us to use the %80 of the data for training and %20 of the data for testing. Also, k-fold cross validation enables us to use different parts of data for training and testing to obtain the best model.

5. References

[1] "The Movies Dataset", *Kaggle.com*, 2018. [Online]. Available: https://www.kaggle.com/rounakbanik/the-movies-dataset/home. [Accessed: 05- Nov- 2018].

[2] F. Ricci, L. Rokach and B. Shapira, *Recommender Systems Handbook*. Boston, MA: Springer US, 2015.

[3] "Machine Learning for Recommender systems — Part 1 (algorithms, evaluation and cold start)", *Medium*, 2018. [Online]. Available: https://medium.com/recombee-blog/machine-learning-for-recommender-systems-part-1-algorithms-evaluation-and-cold-start-6f696683d0ed. [Accessed: 07- Nov- 2018].