Recommender Systems - Netflix

PEP Final Project

Evangelos Tsakeridis Katerina Maxouri
Fivos Tzavellos

2020 January

Abstract

A recommender system is a type of information filtering system that helps predict the preference that a user would give to an item and it is used for recommendations. Its primary use is commercial applications. They have been under development since the early 1990s [1]. Our main focus will be **Netflix**, which is a streaming service that allows members to watch a wide variety of TV shows and movies. First we will present the problem and then the solutions to that problem that are based on machine learning. We will explore how Netflix uses recommender systems and focus, among other things, on algorithms like collaborative filtering and the cosine similarity metric.

Problem

The main product and source of revenue for Netflix is its subscription service. The problem lies in trying to recommend things for people to watch in order to keep them engaged with the product. Netflix differs from the linear broadcast and cable systems in a sense that it gives the user the chance to choose between a wide variety of products. Studies have shown that people lose interest after just one minute of choosing, reviewing 10 to 20 titles. If those titles are not suited for them in a way that they appeal to their interests by being somewhat similar to what they have seen and liked then they abandon the service. According to a McKinsey study [2] 75% of content watched on Netflix are driven by algorithmic recommendations. The company has collected over 1.9 billion ratings from more than 11.7 million subscribers on over 85 thousand titles since October, 1998. It receives over 2 million ratings per day. This volume of data can be used effectively to suggest and recommend content to the user.The common challenge of most of the types of recommender systems is how to deal with massive data to make accurate recommendations. There are three difficulties [3]: (1) the huge amount of data, which requires the algorithm to respond quickly; (2) the sparsity of data, the ratings provided by the users or information which can be used to indicate interests are actually very sparse, compared with the large number of users and items in a recommender system;(3) the dynamic nature of data, which requires the algorithm to update quickly and accurately.

Solutions

Netflix relies on some tools to launch targeted marketing campaigns and make customized suggestions to their users. The tool to make this happen is the recommender system , a collection of different algorithms that constitute the complete Netflix experience. The solution is so effective that there have been cases of people getting addicted to binge-watching (i.e. the process of watching multiple episodes of a television program in rapid succession) and being treated for it [4]. Also another way that Netflix outgrows competition is by putting a lot of money into original and exclusive content creation. They even organized a competition with an aim of improving the accuracy of the rating prediction, resulting in algorithms that we use in production to predict ratings to this day [5]. One of this solutions is collaborative filtering

[6] [7]. CF makes recommendations for the current active user using lots of users' historical rating information without analyzing the content of the information resource.

Relevant Works

There have been some prior works on Recommender Systems based on data given by Netflix itself. Yunhong Zhou et al [8] use a Collaborative Filtering algorithm called alternating-least-squares with weighted - λ - regularization (ALS-WR), which they implement on a parallel Matlab platform. They obtained a Root Mean Square Error score of 0.8985, which is one of the best results based on a pure method. Every algorithm in a recommender system is based on statistical and machine-learning techniques, either supervised (classification , regression) or unsupervised techniques (dimensionality reduction). Such techniques are overviewed by Hastie et al [9] and Murphy [10]. A good introduction to factorization approaches is done by Koren et al [11]. An article by Neil Hunt et al [12] also explores the business value side of the solution , providing some interesting statistics.

Dataset

The first dataset we use is called the Netflix prize data [13]. It contains four files `combined_data_1` until 4 with training data which have the form of

- Movie ID from 1 to 17770 sequentially (as first line of each new movie record / file)
- Customer ID (there are 480189 users)
- Rating (1 to 5)
- Date they gave the ratings (with the format YYYY-MM-DD)

There is also the `movie_titles`.csv file mapping the movie IDs to the movie title and year of release. One file named qualifying.txt that consists of lines indicating a movie id, followed by a colon, and then customer ids and rating dates, one per line for that movie id. The movie and customer ids are contained in the training set. For the Netflix Prize, the program predicted all the ratings the customers gave the movies in the qualifying dataset based on the information in the training dataset. One last file named probe.txt that contains lines indicating a movie id, followed by a colon, and then customer ids, one per line for that movie id. It is used to allow contestants to test their system before they submit a prediction set based on the qualifying dataset. Using this dataset we implement the collaborative filtering algorithm.

The second dataset we use is called `netflix_titles_nov_2019`.csv [14] and it contains data of TV shows and movies on Netflix as of 2019. Its columns contain the unique ID of the show/ film, the title, the directors, the cast, the country, the date when it was added, the release year, the rating, the duration, it's categories, a short description and whether it is a movie or a TV Show. We use it to implement the cosine similarity metric. In order to process it we keep the features for title, director , cast , listed-in (the category) , we fill the NaN values with blank strings and organize the data into rows.

Methods

Cosine Similarity

Cosine similarity is a method to measure the difference between two non zero vectors of an inner product space. Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. The smaller the angle , the higher the cosine similarity. So in this case, what it measures is how similar the preferences between two users are. In our case we use it both for user-user recommendations and for finding items (movies) that are similar to a certain item.

$$\cos(\theta) = \frac{\mathbf{AB}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{A_i B_i}}{\sqrt{\sum_{\mathbf{i=1}}^{\mathbf{n}} (\mathbf{A_i})^{\mathbf{2}}} \sqrt{\sum_{\mathbf{i=1}}^{\mathbf{n}} (\mathbf{B_i})^{\mathbf{2}}}} \tag{1}$$

Collaborative filtering - SVD

The advantages of the use of collaborative filtering for a recommender system are as follows: first, it is independent of the contents of recommended items; second, it can be closely integrated with social networks; third, it has good accuracy in terms of recommendations. **SVD** is a matrix factorization technique commonly used for producing low-rank approximations. Given a matrix X $\in R^{m \times n}$ with rank(X) = r, the Singular Value Decomposition of X is defined as the following:

$$\mathrm{X} = \mathrm{USV}^T \tag{2}$$

where U $\in R^{m \times m}$, V $\in R^{n \times n}$ and S $\in R^{m \times n}$. The matrices U, V are orthogonal, with their columns being the eigenvectors of XX^T and X^TX, respectively. The middle matrix S is a diagonal matrix with r nonzero elements, which are the singular values of X. In our case, U denotes the user features, S is the diagonal matrix of singular values and V transpose are the move features. An important property of SVD, which is particularly useful in recommender system, is that it can provide the optimal approximation to the original matrix X using three smaller matrices multiplication.

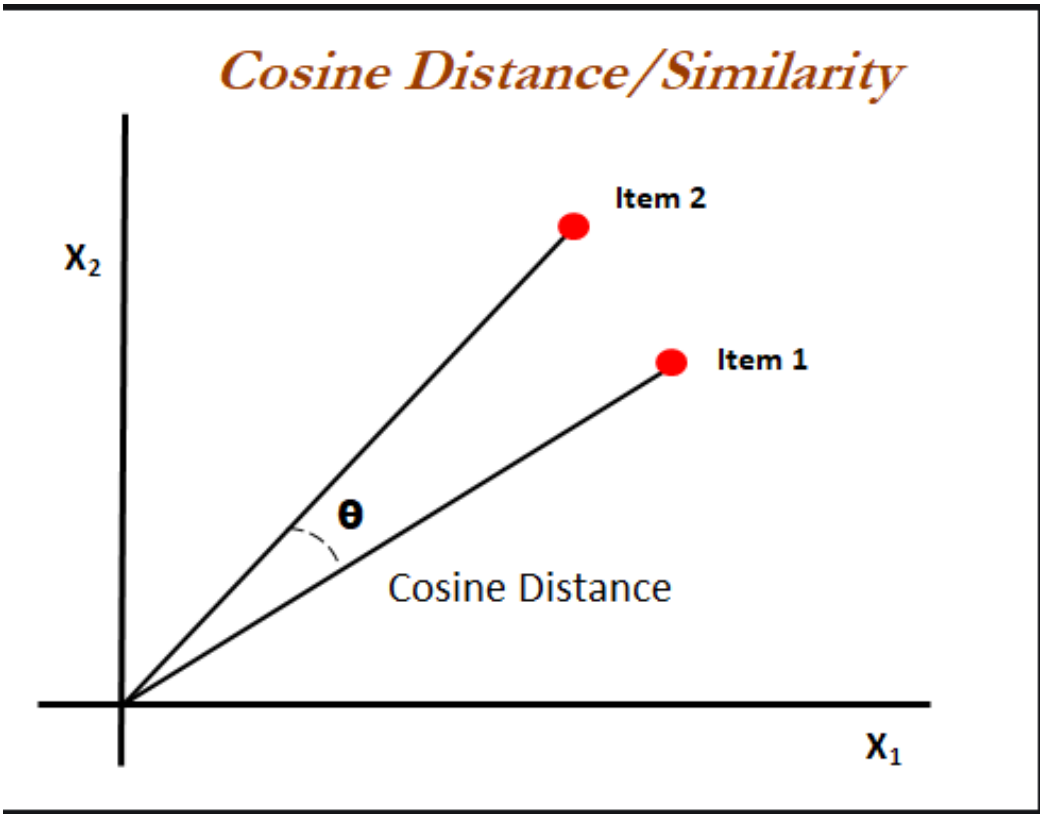

Figure 1: Equation 1 visualized

Mean Rating

Computing the mean rating for all movies creates a ranking. The recommendation will be the same for all users and can be used if there is no information on the user. Variations of this approach can be separate rankings for each country/year/gender and to use them individually to recommend movies/items to the user.It has to be noted that this approach is biased and favors movies with fewer ratings, since large numbers of ratings tend to be less extreme in its mean ratings.

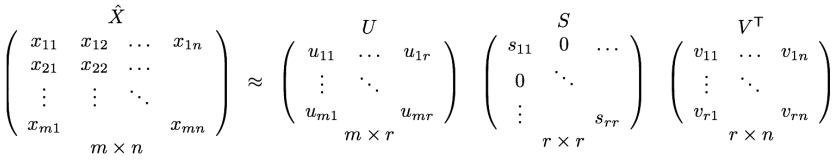

Figure 2: Equation 2 visualized

Weighted Mean Rating

The basic way movies are ranked and recommended would be their rating, but for our case the results are unreliable because the method is too simple and doesn't account for other factors. IMDb's formula is really effective at rating movies [15]

$$W = \frac{Rv + Cm}{v + m} \tag{3}$$

Where:

- W = weighted rating
- R = average for the movie (mean) = (rating)
- v = number of votes for the movie = (votes)
- m = minimum votes required to be listed in the Top Rated list

The variable "m" can be seen as regularizing parameter. Changing it determines how much weight is put onto the movies with many ratings. Even if there is a better ranking the RMSE actually increased slightly. There is a trade-off between interpretability and predictive power.

Cosine TFIDF Description Similarity

TF - IDF (Term Frequency - Inverse Document Frequency) is just a way to measure the importance of tokens in a text. In our case we use it to find similar movies based on their descriptions. TF-IDF adjusts the raw term frequency by taking into account how frequent each term occurs in general (the Document Frequency). Inverse Document Frequency is usually the log of the number of documents divided by the number of documents the term occurs in. TF-IDF is the product of TF and IDF. Using TF-IDF you transform the texts into two real-valued vectors so you can apply the cosine similarity metric on those vectors.

Matrix Factorization as Gradient Descent

Matrix Factorization is the process of decomposing a matrix into (in this case) two matrices, which when multiplied back yields an approximation of the original matrix. In the context of a movie Recommendation Systems, the input X is the ratings matrix, a (number of users x number of movies) sparse matrix. Sparse because most users haven't rated most movies. Matrix Factorization would split them into a pair of matrices M and U of shapes (number of movies x k) and (number of users x k) respectively, representing movies and users respectively. Here k represents a latent variable that encodes a movie or user , thus a movie or user can now be represented by a vector of size k.

As a practical matter, we also want to factor in that people are different and might rate a movie differently, even if they feel the same way about it. Similarly, movies are different, and the same rating for different movies doesn't imply that the rating is identical. So we factor out these biases and call them the user bias b_U, movie (or item) bias b_M and global bias b_G. In order to model this problem as a Gradient Descent problem, we can start with random matrices U and M, random vectors b_U and b_M, and a random scalar b_G. We then attempt to compute an approximation $\hat{\mathrm{X}}$ of the ratings matrix X by composing these random tensors and passing the result through a non-linear activation. We then compute the loss as the mean square error between X and $\hat{\mathrm{X}}$ and then update the random tensors by the gradients of the loss with respect to each tensor. We continue this process until the loss is below some acceptable threshold.

$$X = U * M \approx \hat{X}$$
$$X = U * M + b_U + b_M + b_G \approx \hat{X}$$

Figure 3: Matrix Factorization equation

Deep Learning

Recommendation systems have used deep learning to extract meaningful features for a latent factor model for content-based movie recommendations [16]. Multiview deep learning has been applied for learning user preferences from multiple domains.The model uses a hybrid collaborative and content-based approach and enhances recommendations in multiple tasks [17].

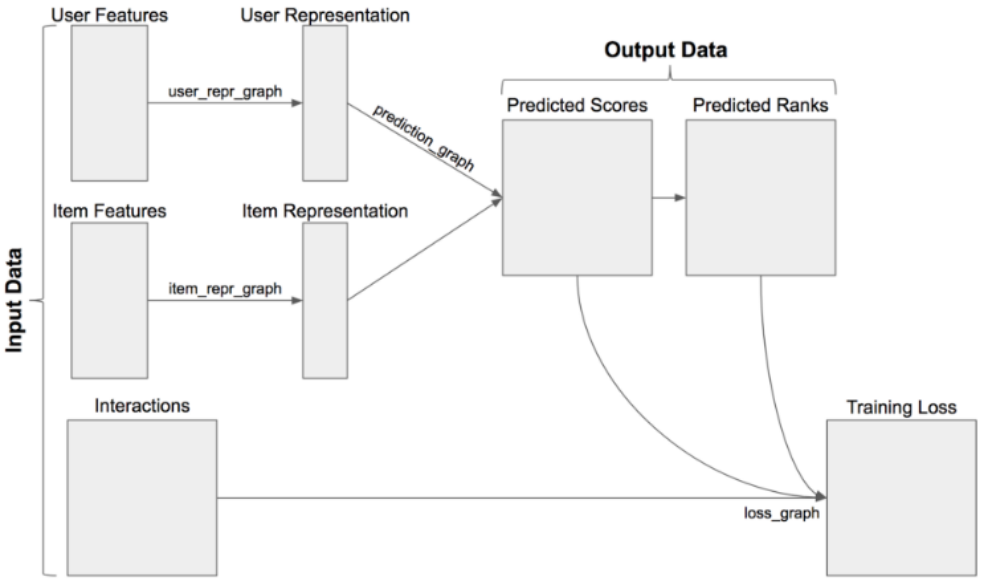

Figure 4: Deep Learning for Recommendation

Conclusion

There are many proposed methods to recommend content to users. Netflix in particular combines some of them in the most efficient way and it actually seems to be a successful attempt, judging by the profits and the economic growth of the company [18]. Based on our results the most efficient method (the one with the highest RMSE) was the User-User Cosine Similarity.

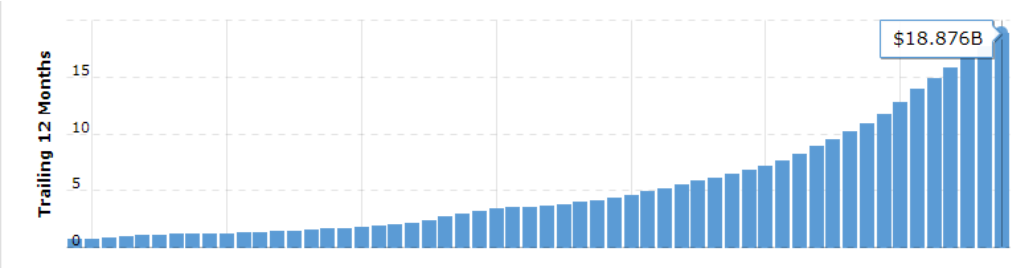

Figure 5: 12 month revenue of Netflix, ending Sept 30 2019

References

[1] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 40, p. 56–58, Mar. 1997.

[2] I. MacKenzie, C. Meyer, and S. Noble, "How retailers can keep up with consumers," 2013.

[3] B. Marlin, *Collaborative filtering: a machine learning perspective*. PhD thesis, University of Toronto, 2004.

[4] B. Gardner, "Three people treated for 'binge watching' addiction to tv in first cases of their kind in britain," 2020.

[5] B. Gardner, "Netflix prize," 2009.

[6] J. H. G. Z. Y. Zhou, Xun; He, "Svd-based incremental approaches for recommender systems," *Journal of Computer and System Sciences*, vol. 81, pp. 717–733, 2015.

[7] R. Chen, Q. Hua, Y.-S. Chang, B. Wang, L. Zhang, and X. Kong, "A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks," *IEEE Access*, vol. 6, pp. 64301–64320, 2018.

[8] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the netflix prize," in *Algorithmic Aspects in Information and Management* (R. Fleischer and J. Xu, eds.), pp. 337–348, Springer Berlin Heidelberg, 2008.

[9] R. T. Trevor Hastie and J. Friedman, *The Elements of Statistical Learning*. Springer, 2011.

[10] K. P. Murphy, *Machine Learning*. MIT Press, Cambridge MA, 2012.

[11] R. B. Yehuda Koren and C. Volinsky, *Matrix factorization techniques for recommender systems*. Computer 8, 2009.

[12] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manage. Inf. Syst.*, vol. 6, Dec. 2016.

[13] "Netflix prize data - dataset from netflix's competition to improve their reccommendation algorithm," 2019. https://www.kaggle.com/netflix-inc/netflix-prize-data/data.

[14] "Netflix movies and tv shows - movies and tv shows listings on netflix," 2019. https://www.kaggle.com/shivamb/netflix-shows.

[15] "How do you calculate the rank of movies and tv shows on the top rated movies and top rated tv show lists?."

[16] X. Feng, H. Zhang, Y. Ren, P. Shang, Y. Zhu, Y. Liang, R. Guan, and D. Xu, "The deep learning–based recommender system "pubmender" for choosing a biomedical publication venue: Development and validation study," *J Med Internet Res*, vol. 21, p. e12957, May 2019.

[17] A. M. Elkahky, Y. Song, and X. He, "A multi-view deep learning approach for cross domain user modeling in recommendation systems," in *WWW 2015*, May 2015.

[18] "Netflix revenue 2006-2019 — nflx."