Paris-Sud University

Do you know your customers?

Authors :
Ghiles Sidi Said
Mohamed Ali Darghouth
Walid Belrhalmia

January 25, 2018

1 Introduction

The aim of this challenge is to predict granular customer/item affinity to support a better decision-making process. In a broader application, this information can drive front-line results in Marketing Campaign or Online Experience Optimization. Traditional approaches usually fail because features do not hold much predictive power. Indeed, most of it resides in the sequence of ratings from users, more than in the user and item descriptions. For this challenge each customer has rated items from 1 (= strongly dislike) to 5 (strongly like). The dataset contains about 2536704 observations, for 92088 users and 3561 items.

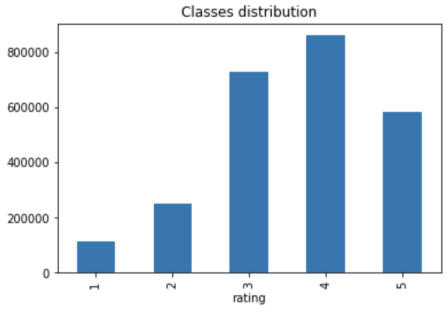

Figure 1: Classes distribution

2 Methodologies

2.1 Basic approach

We transformed the sequence classification problem first to a classical classification by applying a transformation on data.The following figure will illustrate what we did: We applied some classification algorithms on the transformed

Customer	Item	Rating
11676	Item3	2
11676	Item1	5
11676	Item2	3

Customer	Item1	Item2	Item3	Item4	Actual Item	Target
11676	0	0	0	0	3	2
11676	0	0	0	2	1	5
11676	5	0	0	2	2	3

Figure 2: Data transformation

data.

2.2 Matrix factorization

A common approach to solve this type of problems is through matrix factorization, in which we look at the problem as having a set of users and a set of items, and a very sparse matrix that represents known user-to-item ratings. We want to predict missing values in this matrix. In order to do this, we represent each user and each item as a vector of latent features, such that dot products of these vectors closely match known user-to-item ratings.

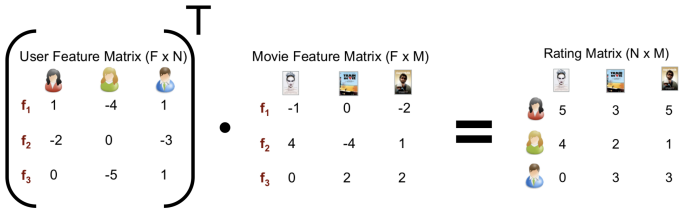

Figure 3: Matrix factorization

The expectation is that unknown user-to-item ratings can be approximated by dot products of corresponding feature vectors, as well. The simplest form of objective function, which we want to minimize, is:

$$\min \sum_{ratings\ u,i} (r_{u,i} - x_u \cdot y_i)^2 + \overbrace{\gamma \cdot \left(\sum_{users\ u} \|x_u\|^2 + \sum_{items\ i} \|y_i\|^2 \right)}^{regularization}$$

Here, r are known user-to-item ratings, and x and y are the user and item feature vectors that we are trying to find. As there are many free parameters, we need the regularization part to prevent overfitting and numerical problems, with gamma being the regularization factor.

It is not currently feasible to find the optimal solution of the above formula in a reasonable time, but there are iterative approaches that start from random feature vectors and gradually improve the solution. After some number of iterations, changes in feature vectors become very small, and convergence is reached.

3 Architectures

First, we started with implementing the factorization matrix using a neural network in keras, we used the architecture below:

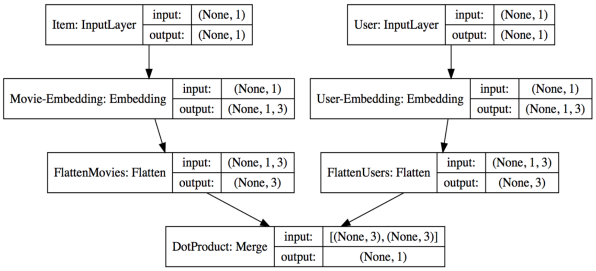

Figure 4: Model using the factorization matrix

Our second approach consisted of using the non-negative matrix factorization by adding a constraint of non-negativity to the matrix of latent features (users and items).

Finally, for our third approach, we used a simple neural network. Instead of taking a dot product of the user and the item embedding, we merge them and use them as features for our neural network. Using that, we are not constrained to the dot product way of combining the embeddings, and can learn complex non-linear relationships. For this model we used the architecture bellow :

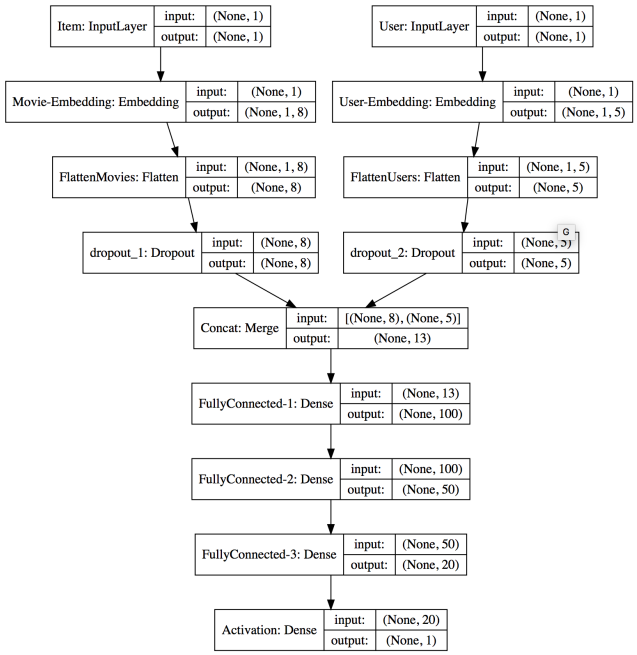

Figure 5: Simple neural network using only the merge

4 Results

RMSE for the different models :

	RMSE
Multinomial	1.5
Logistic Regrssion	1.14
Decision Tree	1.41
Neural network using matrix factorisation.	0.932
Neural network using non-negative matrix factorisation.	0.937
Simple neural network using only the merge.	0.967

Figure 6: Results for different models

The best result we get is 0.932 by using a neural network with a factorization matrix.

5 References

- Recommending items to more than a billion people: https://code.facebook.com/posts/861999383875667/recommending-items-to-more-than-a-billion-people/

- Matrix Factorization Techniques For Recommender Systems:
  https://hpi.de/fileadmin/user_upload/fachgebiete/naumann/lehre/SS2011/Collaborative_Filtering/pres1-matrixfactorization.pdf

- Recommender Systems in Keras:
  https://nipunbatra.github.io/blog/2017/recommend-keras.html