MCMC Method with Ensemble
CS420 Coursework: Item Recommendation

Author
Jinning Li, 515030910592
Shanghai Jiao Tong University
Jinning Li@Computer Science

May 21, 2017

Abstract

Recommender systems are a subclass of information filtering system that seek to predict the "rating" or "preference" that a user would give to an item.

To solve this problem, first of all, a format transforming program was written to transform the initial training data into one-hot encoding.

Then, I optimized the parameters k_0, k_1, k_2, $iter$ to produce many relatively good result. And I employ the ensemble method to make the result more accuracy. I find a weight function and the weighted average seems to be the best.

The model performs pretty well and I get a RMSE of 1.32852, ranking 7. This model can also applied to many other field like Ad. Recommendation and App Recommendation.

1 Introduction

1.1 Background

Recommender systems (RS) or recommendation systems (sometimes replacing "system" with a synonym such as platform or engine) are a subclass of information filtering system that seek to predict the "rating" or "preference" that a user would give to an item.

RS has shown great power to alleviate the workload of users absorbing the massive information on the Internet. Many works are proposed to improve the performance of recommending items e.g. websites, musics, social posts, articles, etc.

In this task, our given data is a ranking from the user to every item. The work include 94263 users and 25634 items. Each record of the data include the user's Id and the item's Id and the ranking's weekday, hour, and two features of items. And of course, the rank.

Then, our task is to construct a model to describe the relation between the features and rank. i.e. to to conduct a comprehensive RS to predict the preference score of the given user on the specific items.

1.2 Data Format

Each line of training data contains:

$$< userId >< itemId >< weekday >< hour >< feature1 >< feature2 >< score >$$

Figure 1: Item Recommendation

Table 1: Notations and descriptions

Notation	Description
ξ_u	The Id of users
ξ_i	The Id of items
ω	The weekday of recording
η	The hour of recording
f_1	Value of feature 1
f_2	Value of feature 2
γ	The Ranking from user

1.3 Notations

Some of the variables in the Report are shown in the table 1

2 Methodology

The method how I use the libFM[2] tool to construct the Item Recommendation model and get the results.

2.1 Data Standardizing

To solve the Item Recommendation problem, the first subproblem is that how to standardize the text data.

The initial data is the record of the ranking. but the form of the initial data is not convenient for us to operate the learning process and not efficient to emphysize the charactoristic feature of each feature dimension.

A reasonable and useful format of training data is the one-hot format. This format is actually a metrix of either 0 or 1. Each row and column represent a feature.

To translate the encoding format, I write a *cpp* program to process the training data and the test data.

Also, for the convenience of parameters adjustment, I part the training data into two parts which contain the data of 80% and 20%.

2.2 Model Choosing

There are many model can solve this recommendation problem, for example:

- SGD, Stochastic Gradient Descent Method
- ALS, Alternating Least Squares Method
- MCMC, Markov Chain Mente Carlo Method
- SGDA, Adaptive Stochastic Gradient Descent Method

In the program, I try all these method in a small scale of training data, and finding that the MCMC method is the most efficient and the easiest to operate. And because we have learnt something about the Bayesian theory, we are more familiar to the MCMC method.

So I choose the MCMC method and go on my work.

2.3 Theory of MCMC method

Many different problems in machine learning such as computer vision, computational biology or recommender systems deal with complex problems in sparse data.

To solve this problem, the Bayesian Factorization Machines is better than the general Factorization Machines in many respects.

In the Baysian sampling the Gibbs sampling is a efficient method. Some modification are applied and we get a faster Gibbs sampler[1].

he Factorization Machines are a regression model for a target y using p explanatory variables $x \in R^p$ as:

$$y(x) = w_0 + \sum_{j=1}^{p} w_j x_j + \sum_{d=2}^{D} \sum_{j_1=1}^{p} ... \sum_{j_d > j_{d-1}}^{p} w_{j1...j_d} \prod_{l=1}^{d} x_{j_1} \tag{1}$$

where second and higher order parameters $wj_1, ..., j_d$ are factorized for $\forall d \geq 2$:

$$w_{j1...j_d} = \sum_{f=1}^{k} \prod_{l=1}^{d} v_{j_1,f}^{d} \tag{2}$$

and the hierarchical Bayesian model adds to the standard regularized regression model Gaussian hyperpriors for each mean μ_θ of all model parameters $\theta \in S = w_0, w_1, ..., w_p, v_{1,1}...v_{p,k}$, but u_{w_0} as well as Gamma hyperpriors for each prior precision α and λ_θ but λ_{w_0}.

$$p(S|y, X, S_H) \propto \prod_{i=1}^{n} \sqrt{\alpha} e^{-\alpha/2(y_i - y(x_i,S))^2} \prod_{\theta \in S} \sqrt{\lambda_\theta} e^{-\lambda_\theta/2(\theta - \mu_\theta)^2} \tag{3}$$

Since posterior inference for FM is analytically intractable we use Gibbs sampling to draw from the posterior. Standard Gibbs sampling is blocked Gibbs sampling. However,

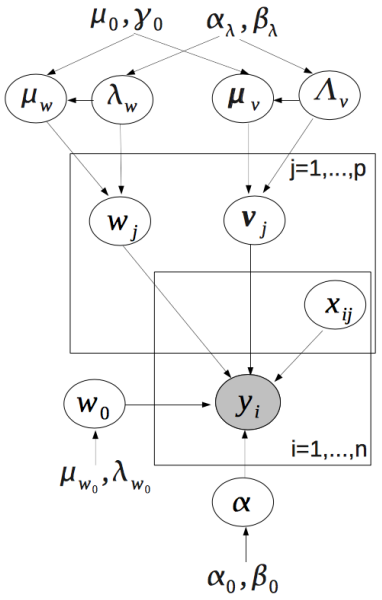

Figure 2: Bayesian Factorization Machines

blocked Gibbs sampling for our hierarchical Bayes approach leads to higher time complexities. Therefore, we propose single parameter Gibbs sampling:

$$y(x|S) = g_\theta(x) + \theta h_\theta(x) \qquad \forall \theta \in S \tag{4}$$

Using this notation and parameters $S_0 = \{\gamma_0, \mu_0, \alpha_0, \beta_0, \alpha_\lambda, \beta_\lambda\}$ of the conjugate hyperpriors the conditional posterior distributions for each θ, μ_θ, λ_θ, and α resulting from our hierarchical Bayesian model are:

$$p(\theta|y, X, S\backslash\{\theta\}, S_H, S_0) = N(\mu_0^*, \sigma_\theta^2) \tag{5}$$

$$p(\lambda|y, X, S, S_H\backslash\{\lambda_\theta\}, S_0) = \Gamma(\alpha_\theta, \beta_\theta) \tag{6}$$

$$p(\alpha|y, X, S, S_H\backslash\{\alpha\}, S_0) = \Gamma(\alpha_n, \beta_n) \tag{7}$$

$$p(\mu_0|y, X, S, S_H\backslash\{\mu_\theta\}, S_0) = N(\mu_{\mu_\theta}, \sigma_{\mu_\theta}^2) \tag{8}$$

2.4 Training

In the subsection 2.1, we have produce the feature map with the one-hot encoding.

In libFM, the tool provides a convenient way to training my model. There are four parameters to optimize:

- k_0, whether global bias method is used.

- k_1, whther one-way interactions is used.
- k_2, the number of factors used for pairwise interaction.
- *iter*, the number of iterations.

To optimize these parameters, I use the parted training data with the percentage of 80% and 20% to test. And I find opening the global bias and the one-way interactions can increase the accuracy of the result. So I let $k_0 = 1$ and $k_1 = 1$.

For k_2, in the beginning, I think that the number of factors should be as large as possible. But the experiment proves I am wrong. The output's changing with the k_2 is a convex function. i.e. a value in the middle is better than in the boundary. I think this is because overfitting when k_2 is large. After a lot of trying, I find that $k_2 = 70$ around is the best.

For *iter*, I think it actually should be as large as possible. but after the Markov Chain become steady, the iteration makes no sense but cause the running time to long. And I find that the larger k_2 is, the more iterations will it take to make Markov Chain steady.

And for the ensemble process, I produce a lot of output data of $k_2 = 25$ to $k_2 = 125$ with suitable *iter*.

2.5 Ensemble

For the Predict Process, we also Standardize the testdata as is illustrated in the section 2.1.

Then, use the tool libFM, we produce the Result of different k_2 and *iter*.

To increase the accuracy of the result, I use the ensemble method to strengthen the result. I try the average and weighted average, and the R.M.S. method, I find the weighted average is the best one and the most flexible. I set the relative weigh as:

$$w_i = 1 - |k_{2,i} - 70|/70 \tag{9}$$

After weighted average, the result have a large improvement.

3 Discussion and Future works

3.1 Discussion

- The MCMC method can excellently solve the RS problem.
- The MCMC method takes a lot of iteration to converge.
- The calculation takes too many time so that many training didn't finish before deadline.
- The Final score is 1.32852, which is really good and pass the strong baseline.
- If time permits and other method is added to the ensemble, I think the result will have a improvement.
- And because the large calculation scale, I learn to use the online server with ubuntu system. That's interesting.

3.2 Future works

The training process should be optimized. SGD, ALS and some other method can be taken into considering when ensembling. This Item Recommendation model can also be used to solve the advertisement, and App recommendations. And a problem to solve is that if the dimension of feature is pretty large, the runtime efficiency shoule be taken into considering.

References

[1] C. Freudenthaler, L. Schmidt-thieme, and S. Rendle. Bayesian factorization machines.

[2] S. Rendle. Factorization machines with libFM. *ACM Trans. Intell. Syst. Technol.*, 3(3):57:1–57:22, May 2012.