A Solution to the EMI Music Data Science Hackathon

linus
Team "lns"

1 Abstract

My solution to the EMI Music Data Science Hackathon is a combination of libFM[1] with gbm[3]. Firstly, several latent factor models are fitted to the target. Then a gbm is fitted on the residuals of the linear combination of the factor models. An improved version of my approach is also documented, which scores below 13.2 on both public and private leaderboard.

2 Latent Factor Model

Latent factor model has been proved very powerful as a recommendation system in many applications. The latent factors can model the different characteristics of items and user preferences very well. In this task, a well-written software libFM[1] is used to fit this latent factor model.

Firstly a simple setting is used for the factor model. We extract User, Artist and Track features from `train.csv` and `test.csv` as below:

For each artist, track and user, we have a seperated column indicating whether this rating is from this user (of this track or of this artist). Thus, each row in the data matrix contains exactly 3 '1's, representing artist, track and user respectively.

The main idea of latent factor model is to capture the interaction between users and tracks. Roughly speaking, tracks can be categorized into several

Figure 1: Data File used in libFM (train.libfm)

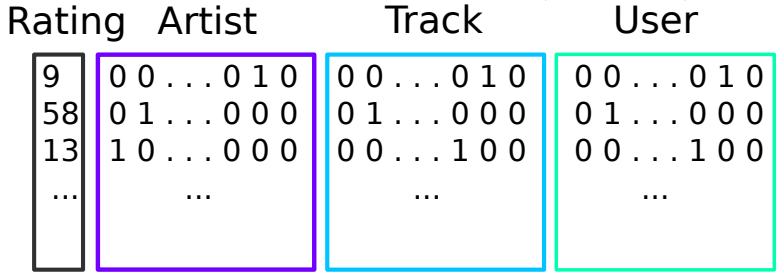
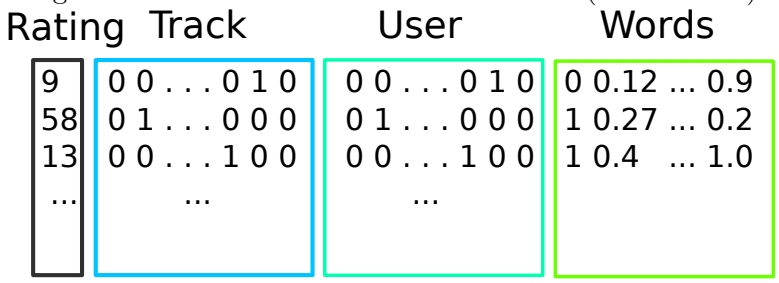


Figure 2: Another Data File used in libFM (train.libfm3)



groups and so are users. And users in the same group are tend to have similar taste of different kinds of music. The latent factors are used to model users' different preferences of music from different categories. This approach has been proved successful in many past applications.

I used libFM to train this part of model. libFM is a very powerful and easy-to-use tool for machine learning researchers and engineers. SGD, ALS and MCMC are tried on the first data file (train.libfm). After the competition, I also tried something like libFM's author Steffen's approach posted in the competition forum[1]. Track columns and user columns are combined with columns from `words.csv`. A MCMC model is trained with this data file (train.libfm3).

Name	datafile	method	dim	iter	init stdev	pub score	prv score
submit.mcmc	train.libfm	mcmc	1,1,8	500	1	15.16819	15.18699
submit.als	train.libfm	als	1,1,8	200	0.5	15.29393	15.31992
submit.sgd	train.libfm	sgd	1,1,12	1000	0.5	15.25401	15.26853
submit.mcmc2	train.libfm3	mcmc	1,1,8	1000	0.2	13.26927	13.31100

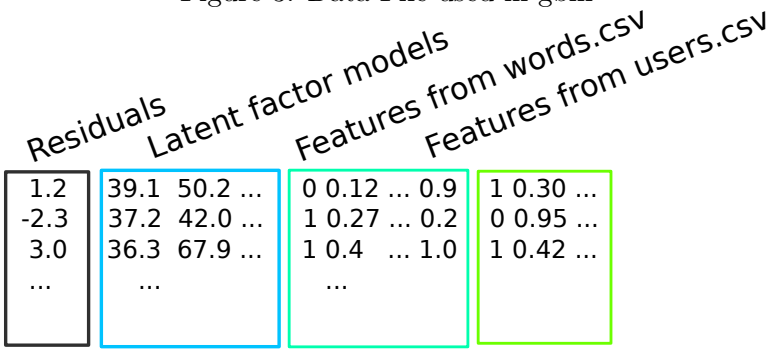
3 Boosting Trees

Friedman's Gradient Boosting Machine[2] is a very powerful model for general purposes. It can capture non-linear relationship and complicated relations between several features very well. After the latent factor models are trained, a linear regression is fitted, and boosting trees are trained to fit the residuals of the linear regression. I used `lm` and package `gbm` in R for this part of model.

Name	latent factor model used	iters	depth	shrinkage	minobsinnode	score
submit.gbm1	mcmc,als,sgd	2000	6	0.02	3	≈ 13.4
submit.gbm2	mcmc,als,sgd,mcmc2	2000	6	0.02	3	≈ 13.1

[1] `http://www.kaggle.com/c/MusicHackathon/forums/t/2242/code-approach-sharing/12821#post12821` My parameters for submit.mcmc2 are a little different with this.

Figure 3: Data File used in gbm



4 Further discussion

From the above, we can know that improvements on the latent factor model would like to lead to a better final score. Also `n_m` has suggested training `gbm` for each artist seperately and reported big improvement on the result.[2]. Generally a combination of different models would have a boosting effect on the result. We are hoping to reach a better score (probably 13.0 or less?) with all people's efforts.

References

[1] Steffen Rendle: **Factorization Machines with libFM**, `http://dl.acm.org/citation.cfm?id=2168771`

[2] Friedman, J. H. : **Stochastic Gradient Boosting**, `http://www-stat.stanford.edu/~jhf/ftp/stobst.pdf`

[3] Greg Ridgeway : **Generalized Boosted Models: A guide to the gbm package**, `http://cran.r-project.org/web/packages/gbm/vignettes/gbm.pdf`

[2] `http://www.kaggle.com/c/MusicHackathon/forums/t/2243/what-was-your-single-model-performance/12646#post12646`