Overview of Classic and Popular music recommendation algorithms

Fan Wu

November 7, 2021

1 Introduction

Music recommendation algorithm is one of the most common algorithms we might use in our daily life. Most music software, such as Spotify, YouTube music adapt such algorithm to help users to find music they might be interested. Therefore, the performance of music recommendation algorithm has become a major concern in the industry. This review compares the performance between popular music recommendation algorithm and classic recommendation models based on a Kaggle music recommendation competition: WSDM - KKBox's Music Recommendation Challenge. The competition is looking for an effective recommendation algorithm which predict whether a user will repeatedly listen to a song based on the given training set.[2]
One of the most widely used recommendation algorithm among the submitted source code is Light GBM. Whereas the original algorithm used by KKBox is a collaborative filtering based algorithm with matrix factorization and word embedding.
The review will analysis a popular Gradient boosting framework: LightGBM, and one of the most classic classification model: k-nearest neighbors algorithm. Moreover, the review is going to evaluate their performance based on the prediction accuracy. Besides, the review will be more focused on the perspective of application using Light GBM, since it's a widely used boosting framework.

2 Methodology

In the dataset provided by the sponsor, there is an attribute:"target value" in train table which indicate whether a song is repeated by the user. Therefor, the algorithm should output the predicted target value based on the existing text information of the user and songs.

2.1 k-nearest neighbors algorithm

As one of the simplest classification algorithm, KNN is relatively powerful. The basic idea of KNN is to treat every table entry as a data point. Based on the known data points in training set which had target_value = 1, when unseen new points come in. It will predict the target value by choosing the top K points which near to the point

which target value = 1, it will simply predict these k points also has target value = 1, and vice versa.
Since KNN is a such simple and intuitive algorithm, it inevitably has some flaws. KNN is a relatively slow algorithm based on the dataset size because it needs to calculate the distance between two points for all the data entries. Besides, if there are too many features involved in the model, then by the curse of dimensionality, KNN algorithm is straggle to predict the target value. Moreover, KNN is pretty sensitive to the outlier values and missing values which often exists in the dataset.[3][4]

2.2 LightGBM

LightGBM(Light Gradient Boosting Machine) is a gradient boosting framework that uses decision tree based learning algorithm, designed by Microsoft. Gradient Boosting Decision Tree is a popular model used in industry. It usually implemented to solve about multi-classification, click-through rate prediction, search ranking problems.
As a framework for implementing the GBDT algorithm, LightGBM support efficient parallel training, and it has the advantages of faster training speed, lower memory consumption, better accuracy, support for distributed processing of massive data quickly and so on. The idea of LightGBM is it grows the decision tree vertically while other algorithm grow trees horizontally, which means LightGBM grows tree leaf-wise while other grows level-wise.[1][5]
The advantage of Leaf-wise is that, with the same number of splits, Leaf-wise can reduce more errors and get better accuracy; the disadvantage of Leaf-wise is that it may grow a deeper decision tree and produce over-fitting. So LightGBM adds a maximum depth limit to the Leaf-wise, ensuring high efficiency while preventing over-fitting.

3 Comparison

For the given dataset, there exist NaN value and outlier value for users' age. For simplicity, I treat "NaN" value as an independent category and set the range of user age from 10 to 75, any value outside the range will be treated as unknown.
There also exist some useless features inside the song table and user table, so I removed couple of them to reduce the data dimensionality for KNN algorithm.
For KNN algorithm, I set k = 20, and the AUC score for k = 20 for test set is around 0.658 whereas the LightGBM output value around 0.748.
Since both model based on the same processed data, there is no difference between the feature engineering part. We can see that LightGBM has a better performance compared to KNN algorithm.

3.1 Conclusion

The main reason for this is that the problem of predict the target value in dataset is more similar to a click value rate problem. We can treat repeat a song as an event equal to the user click into an ad. Therefor, LightGBM takes the advantage to solve it, while KNN is used to solve neighborhood based collaborative filtering problem. KNN also has a disadvantage in handling large dimensional data, which definitely decreases

the accuracy of prediction.
Overall, for the typical recommendation problem which want to recommend similar items, KNN algorithm can be considered as a baseline module because of it's constraints. LightGBM as a popular and mature boosting framework can provide a faster training speed and better results, but feature engineering is also an essential part since it can directly be related to the prediction accuracy for specific problem.

References

[1] Mandot, P. (2018, December 1). What is LIGHTGBM, how to implement it? how to fine tune the parameters? Medium.

[2] KKBOX. "WSDM - KKBox's Music Recommendation Challenge" https://www.kaggle .com/c/kkbox-music-recommendation-challenge

[3] Genesis. (2018, November 25). Fromthegenesis.com - pros and cons of K-nearest neighbors.https://www.nichesitemastery.com/site/fromthegenesis.com.

[4] Gongde, G. "KNN model-based approach in classification." OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". Springer, Berlin, Heidelberg, 2003.

[5] Guolin, K. "Lightgbm: A highly efficient gradient boosting decision tree." Advances in neural information processing systems 30 (2017): 3146-3154.