Machine Learning

Final Project

Team SDJ

R04942023 Chih-Hsiang Liou R04942045 Da-Min Huang R04942055 Yi-Hsun Lin

Graduate Institute of Communication Engineering, National Taiwan University

January, 20, 2016

1 Feature Extraction and Blending

By using sample training set, the scores of using any one of following models is not bad. But to enhance our performance, we find out that the adjustment of parameters helps a little. New features is the key to stands out in the competition.

1.1 Final Scores

	private scores	rank
Track 1	0.970150	10
Track 2	0.889089	14

1.2 Using Data from Log Files

We noticed that followings features are lost in sample train set.

1. Number of objects

   We was trying to use the object and `object.csv` to find some more features. But there are many objects not recorded in `object.csv`, so we then calculate the

number of object each ID has taken and add two features combine with sample train set. One is number of objects taken by ID, the other is by some user.

The number of objects taken by ID is the 4^{th} important feature in our train set. This helps a lot.

2. Time

   We find that there are no features considering time property in sample train set. So we gathering the loging time of each log and get

   (a) The time between last log and first log

   (b) After the course announced, how many times one user log in to the course in each day. From this we get 30 more features from 1^{st} day to 30^{th} day.

   After adding all these features about time into our train set, we find out that the E_{val} of Gradient Boosted Tree model from 0.13... to 0.12... and the we make a lot improvement.

1.3 Blending for Final Output

1.3.1 For Track 1

Due to the evaluation policy of Track 1, the most important is to make the ID with high probability of droping course must be in the top 9000 enrollments. Hence we make apply the following rules to the result predicted by Gradient Boost Tree model.

1. If the probability of droping course is larger than 0.9, compare the result obtained by other two models. If the average of other two results is more than 0.8, adjust the probability to be 1.0.

2. If the probability of droping course is smaller than 0.1, compare the result obtained by other two models. If the average of other two results is less than 0.2, adjust the probability to be 0.0.

Hence we make sure that top 9000 courses are with high probability of droping courses.

1.3.2 For Track 2

We simply apply uniform blending to the best results obtained by three models. If some enrollment got two or more 1 outputs, then output 1; got two or more 0 outputs, then output 0.

2 Using Models

2.1 Random Forest

2.1.1 Pros of Random Forest

1. efficiency

   Decision Tree itself is very efficient;whats even better is that one can train every tree in the forest using multiprocessing tool simultaneously.

2. scalability

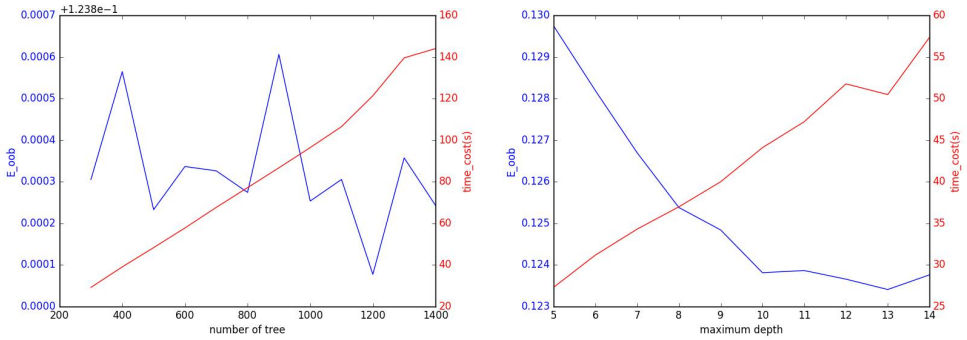
   If the training data has n samples and d features,the computational cost of training a "nonrandom" forest of M trees is $O\left(M\left(n \times d \times \log n\right)\right)$. Although the application of randomness will increase the complexity,it still provides a good estimate of the time spent during training.This estimate tells us that the time spent for the algorithm will increase at an acceptable rate as n increases, which means that RF is a good model under the condition of large-scale data.

3. interpretability

   Because of the use of Decision Stump algorithm, one can easily check the chosen feature and the threshold of each node ,which reveal the reason why Random Forest makes certain decision.As a result, it has good interpretability.

2.1.2 About the Parameters in RF

The main two parameters needed to be adjusted are number of trees and the regularization of trees.



(a) number of trees (b) maximum depth

Figure 1: out-of-bag error versus time spent

1. number of trees

From figure (1a) we can find that as the number of trees increases, the time spent during the training increases steadily but the out-of-bag error goes up and down. Consequently, there is no need to use too many trees, and we choose to use 1000 as the number of trees.

2. regularization of trees

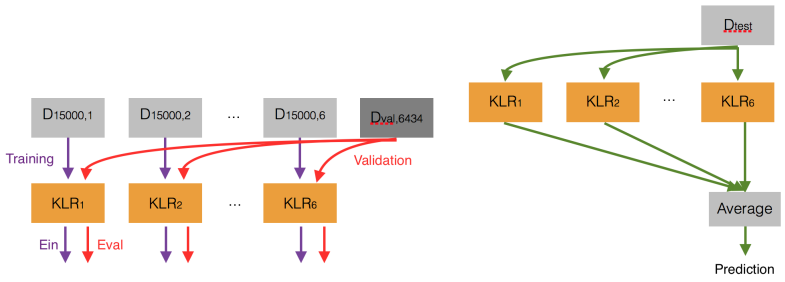
   From figure (1b) we can find that as the maximum depth increases, out-of-bag error decreases and time spent during the training increases. However, because the growing rate of time is higher than the decreasing rate of out-of-bag error,we choose to use 10 as the maximum depth of each tree.

2.1.3 About the Use of Training Data

Because we have lots of features in our training and testing data(55 in total),we decide to remove those unimportant features during training. First,we use the `feature_importances_` function in RandomForestClassifier of SKLearn to get the relative importance of each feature; then we check the performance of Random Forest after removing some unimportant features and decide the optimal number of removals, according to the attribute `oob_score` in Random-ForestClassifier.We discover that removing the two least important features will increase the performance of the classifier.

2.2 Uniform aggregation of Gaussian Kernel Logistic Regression

In this final project, we also try kernel logistic regression as the model as consideration. Since we aim to find an method that can predict the probability as target for effetely and large model complexity, we choose logistic regression with gaussian kernel as our model. However, in order to reduce memory usage and computation time, we divided the training data into 6 part, and trained 6 GKLR (Gaussian Kernel logistic regression separately), then use the uniform aggregation to blend them all.
The model are shown in following diagram. First one is the training phase and the second one is the prediction phase. We use 90000 data as training data and 6434 data as validation data, and each KLR are training with 15000 data separately.

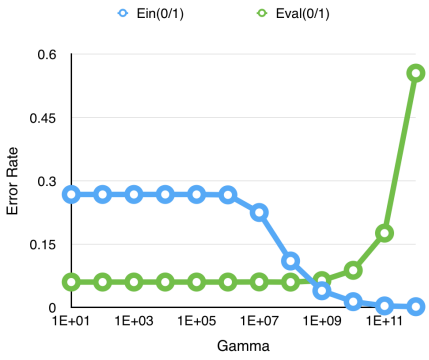


In the prediction phase, the 6 KLR models are aggregate uniformly to prediction an probability. Note that, we have to decide two parameter is this model. First is the lambda constant for the L2-regularizer penalty, Second is the gamma constant of the gaussian kernel. We try different parameter and select the best one of good validation error.

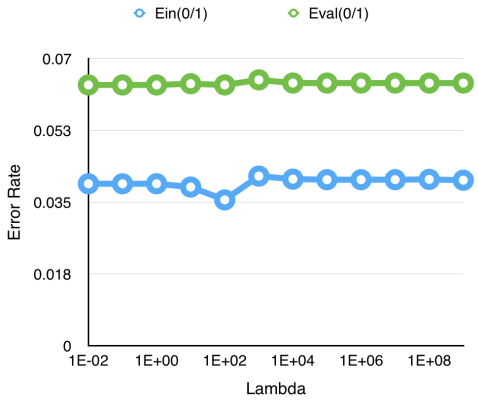
2.2.1 Equation

$$\min_{\beta} \left[\lambda \boldsymbol{\beta}^T \mathbf{K} \boldsymbol{\beta} + \sum_{n=1}^{N} \log\left(1 + \exp\left(-y \otimes \mathbf{K}\boldsymbol{\beta}\right)\right) \right], \ \mathbf{K}_{m,n} = \exp\left(-\gamma \left\|\mathbf{x}_m - \mathbf{x}_n\right\|^2\right) \tag{1}$$

2.2.2 Validation Result : (with 55 features)



(a) Changing γ



(b) Changing λ

2.2.3 Discussion

From figure above, we can see that with gamma equal 10^9, the model have less validation error. Also, with Lambda equal 10^2, the model have less in-sample error. As a result, we choose gamma to be 10^9 and Lambda to be 10^2.

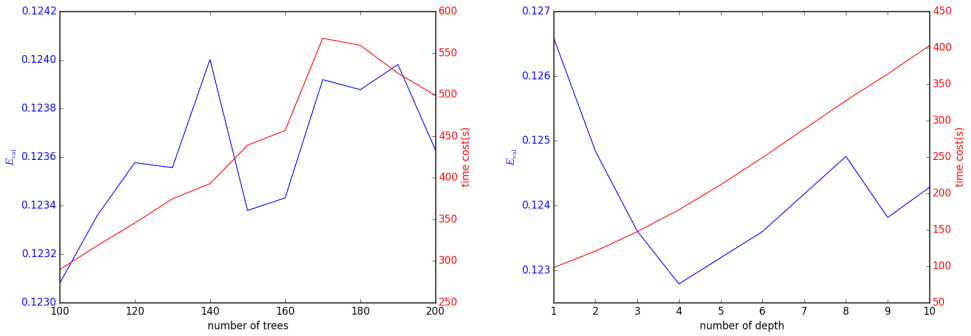
2.3 Gradient Tree Boosting

Here I used the `sklearn.ensemble.GradientBoostingClassifier`.

2.3.1 Pros of Gradient Tree Boosting

1. Efficiency
2. Predictive power
3. Robustness to outliers in output space

2.3.2 How to Choose Parameters



(a) number of trees (b) number of depth

Figure 3: E_{val} v.s. time costs

From figure (3a) we see that E_{val} number of trees grows as number of trees increases. So we choose 100 to be the number of trees.
From figure (3b) we can see that the best E_{val} is around 4 depth, we choose depth to be 6 finally.

2.3.3 Some Other Pre-processing Before Training

By using the attributes `feature_importances_` of this model, we removed first two not important features to enhance our performance. This attribute will show the score of each feature. After removing the features, we start to train the model.

Reference

[1] Lecture Notes by Hsuan-Tien LIN, Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan.

[2] sklearn: `http://scikit-learn.org/`

[3] TensorFlow: `https://www.tensorflow.org/`