CS-411 Project Report

Meryem M'hamdi
EPFL
meryem.mhamdi@epfl.ch

Nevena Radovanovic
EPFL
nevena.radovanovic@epfl.ch

Abstract

This work aims at predicting grade improvement in problem submissions based on certain patterns in video and forum activities in the course. In this work, we dig deep into the analysis of the MOOC dataset, study correlations of different subsets of both existing, new and aggregated features with the grade difference, fit different machine learning classifiers and models and fine tune their parameters to increase their performance. We will present in what follows the rational used to extract and select a subset of features that when fed to different algorithms give us the best possible results. We argue that Random Forest is the best algorithm in this case as it gave us improved performance of 64% on the test dataset.

1 Exploratory Data Analysis

1.1 Data Description

The training dataset X_train consists of 42954 rows where each record corresponds to 1 submission of a particular student in a specific problem. When students submit more than once, the events in the clickstream represent all video and forum events they have taken in between submissions. In this task we are asked to predict the grade improvement in a submission based on the actions undertaken by the students since the previous submission. So, we define a binary variable improved based on GradeDiff and use it as label in our classifiers.

The testing dataset X_test consists of 26062 rows structured in the same way as training dataset, however with no grade information provided.

The initial data table contained: ProblemID, UserID, SubmissionNumber, TimeStamp, TimeSinceLast (time since last submission), Grade, GradeDiff, NVideoEvents (number of video events: plays, loads, pauses, etc.), NForumEvents (number of forum events: comments, posts, thread views, etc.). In addition, several custom features described in Section 2.1 were given.

1.2 Data Visualization, Cleaning and Pre-processing

1.2.1 Dealing with Nans

Out of 42954 submission records, 31985 have gradeDiff information, so we keep only those. This automatically discards submission records with submission number 0 since there is no information about grade difference. We also exclude records where neither video nor forum events were reported, leaving us with a total of 4948 data points (which is roughly 10% of the whole raw MOOC dataset).

1.2.2 Dealing with Class Imbalances

We observed that the train dataset consisted of unbalanced distributions of classes. The number of submission records with NO grade improvement is approximately 65% which could result in high bias during model fitting. If we don't undertake some measure for classes rebalancing, our

classifiers will do a better job finding true negatives (NOs) versus finding true positives since we don't feed them with enough data to learn positive cases. For this reason, we create similar number of submissions records for each class using upsampling.

1.2.3 Data Scaling and Normalization

We observed that the different variables used exhibit high variance. Some of the algorithms we are going to use are sensitive to magnitude and can converge far faster on normalized data. In order to avoid biasing our classifier to those variables that have high variance, we normalized the data using preProcess function with center and scale options.

2 Feature Extraction and Engineering

2.1 Existing Features

To create custom features that could be better predictors, specifics of each video and forum event are used. For each video event (load, play, download, seek, pause speed change, error, stale) information about TimeStamp, EventType, VideoID, CurrentTime, OldTime, NewTime, SeekType, OldSpeed, NewSpeed is given. Similarly, for each forum event (load, subscribe, thread subscribe, thread launch, thread view, thread post on, post upvote, post downvote, comment downvote) the information about TimeStamp, EventType, PostType, PostID, PostLength is available.

As mentioned, the initial set of features contained several custom features.

Video related custom features:

- Time difference between last and first video activity:
  DurationOfVideoActivity = TimeStamps[-1] - TimeStamps[0]
- Average time between two video events:
  AverageVideoTimeDiffs = sum(TimeStampDiffs)/max(1,len(TimeStampDiffs))

Forum related custom features:

- NumberOfThreadViews = EventTypes.count('Forum.Thread.View')

2.2 New Features Rationale

In order to set clear hypothesis about what helps students to improve their grade, the first step was to perform a visual examination of the data as well as a literature review (1), (2), (3), (4). The preliminary analysis brought us to the following assumptions:

- Hypothesis 1: The greater the number of forum and video activities, the greater the chance for the grade improvement.
- Hypothesis 2: The weight of activity matters. Forum activities located deeply in the forum hierarchy (posts, comments) can imply a greater involvement of the student, while simply viewing the threads is not necessarily an indicator of student's actual willingness to reflect on and understand the concept in question.
- Hypothesis 3: The more submissions a student makes, the greater the chance of improvement. We presume that, after each submission, student reflects on mistakes in his understanding and identifies problems which he tries to overcome before the next submission.

In accordance to our first hypothesis we created several simple features that count a number of each event type (NumberOfPlays, NumberOfDownloads, NumberOfPauses, NumberOfLoads, NumberOfSpeedChange, NumberOfThreadViews, NumberOfComments, NumberOfPosts, NumberOfThreadsLaunched).

In addition, second hypothesis was supported by aggregate features ScoreRelevantEvents and EngagementIndex. The idea of ScoreRelevantEvents feature was to weight different types of forum events depending on their position in forum hierarchy. It is a sum of NumberOfComments, NumberOfPosts and NumberOfThreadViews with a weight of 2,1.5 and 1 respectively. EngagementIndex

is calculated as TotalVideoEvents * TotalForumEvents. The idea is that having the combination of video and forum events can help in improving the grade as it reflects an overall engagement of the student.

Finally, the third hypothesis incited us to add SubmissionId which reflects number of submissions that a student had before the one in question, as well as SpeedOfSubmission. The later one is calculated as: SpeedOfSubmission=submission timestamp/(1+ total number of submissions). It is added in order to account for rapid, successive submissions in between which a student cannot make any progress in learning.

As the initial set of features was not a good predictor, we added more features (new or aggregated from existing ones) based on different assumptions:

- DistinctIds - Number of distinct video ids in video activities. The reasoning is that we should distinguish between 5 plays/pauses of one video and plays of 5 different videos.
- SeenVideo - If a student saw a video or not. The idea is to distinguish between, for example, only having video loads and having video plays, as in the first case student could not make a learning progress. We assume that a student saw a video if there is a play (watched online) or download (watched offline) event.
- IsLastSubm - If the submission in question is the last submission that a student made for a certain problem. Similarly to our third hypothesis, we suppose that the last submission is the best one, improved from previous times.
- TotalTimeVideo - Calculated as DurationOfVideoActivity*DistinctIds. The idea is to highlight the importance of having the combination of time invested in watching videos and watching multiple videos.

In order to select the best features, we computed Pearson correlation coefficients between aforementioned features and "improved" status. Unfortunately, none of the feature did not seem to have significant correlation. By exploring different combinations, and taking into account small differences in correlations, the set of features that led to best prediction is the following:

SeenVideo, NVideoAndForum, IsLastSubm, NumberOfThreadsLaunched, SubmissionNumber, TotalTimeVideo, EngagementIndex

As this set of features gave the best accuracy we can argue that:

- As we supposed, the total number of activities matters. By being active a student increases his chance to improve the grade. Furthermore, watching at least one video seems to be preferable as well.
- Weighted sum of different forum events wasn't a good predictor (probably because of inappropriate weights). However, having the NumberOfThreadLaunch as a separate feature led to better accuracy, which implies that certain activities (as thread launching) have greater weight then others as supposed in Hypothesis 2.
- The more the student submits, the greater the chances of improvement. The grade of last submission is usually an improvement from previous one. SpeedOfSubmission did not give good predictions as the dataset doesn't contain a lot of students having very short time between submissions.
- The time that a student invested in MOOC activities is important for improving the grade.
- Combination of video and forum events is important. Watching videos is not as effective as when combined with forum activities (and vice versa).

2.3 Covariance Matrix

To visualize the correlations between different variables, we plot the covariance matrices in figure 1. It shows some hidden relationships between variables that have low to inexistant correlations.

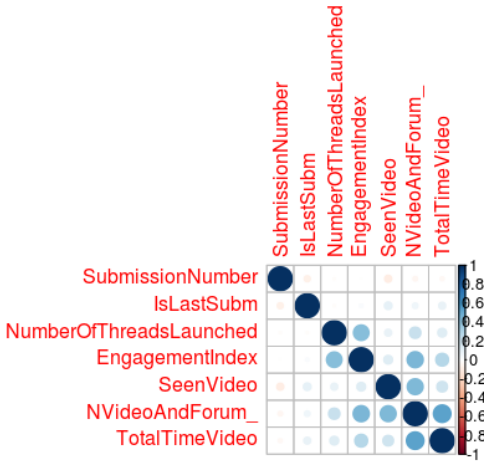

Figure 1: Correlation Matrix between the different variables

2.4 Dimensionality Reduction

Because of the curse of high dimensionality, fitting machine learning models with more dimensions tend to result in less predictive power. We tried applying dimensionality reduction technique to keep only a few principal components that capture most of the variability of the data. For that purpose, we used PCA which is a linear orthogonal transformation that transforms the data to a new coordinate system. We tried with different numbers of principal components (2,3,4), but the results both on training and kaggle submissions didn't encourage us to pursue this path so we decided not to consider this technique, but rather work on selecting from the beginning few meaningful features that are better correlated with grade predictions.

3 MultiVariate Outlier Detection and Removal

After visualizing data, we have removed outliers to reduce skew in the data set. Outliers can hugely mislead the training process of some machine learning algorithms resulting in over-fitting and poor accuracy results. We use package 'mvoutlier' for that purpose.

4 Training Classifiers

To come up with the best prediction model possible that not only gives good training results but generalizes well on the testing dataset, we fed our data into several classifiers. We also worked carefully on proper parameter tuning by performing cross validation of different split permutations of the training dataset and several repetitions to optimize the cross-validated training error. We were careful enough to avoid underfitting and overfitting by observing learning curves and how more data affect bias and variance. In what follows, we describe in more details the algorithms we have tried, their suitability to the task in hand, the different parameters used to tune them, how we used them to learn about feature importance, and evaluation methodology using both learning curves and confusion matrices.

4.1 Binomial Logistic Regression

We started our binary classification by applying the most simplistic and naive strategy which consists of fitting a baseline to the data using logistic regression. We used 4-fold cross validation to separate our data into train and test data and we trained for different values of parameter alpha.

4.2 K-Nearest Neighbors

K-Nearest Neighbors can be considered as another lazy baseline due to its simplicity whose efficiency depends on stored labels against which the new testing dataset is compared using some similarity measure. The parameter that was fine tuned here is the number of neighbors which we want our classifier to base classification on by doing majority voting. It turns out that 2 neighbors gives the best cross-validated results, so we use this version of knn in our comparison with other methods.

4.3 Random Forest

Random forest is a tempting technique worth trying. It not only reduces the variance of decision trees which lowers over-fitting and leads to increased performance but also is faster and scalable. Applying this method allowed us to base our classification prediction on a set of trees instead of one decision tree by performing a majority voting that is most likely to be more accurate than the results of single trees. Its strength comes from its ability to perform feature engineering on its own applying different weights depending on the importance of the features.

4.3.1 Learning Curves

To visualize how the classifier performs as we increase the size of data and to determine whether we are suffering from a model with high variance or high bias, we plot the learning curves for different splits of the training dataset. As shown in figure 2, random forest classifier benefits from adding more data as the testing performance increases towards the end and the variance between the training and testing shrinks (low variance and low bias). So, as more data is added, we observe reduction of overfitting and better predictions.

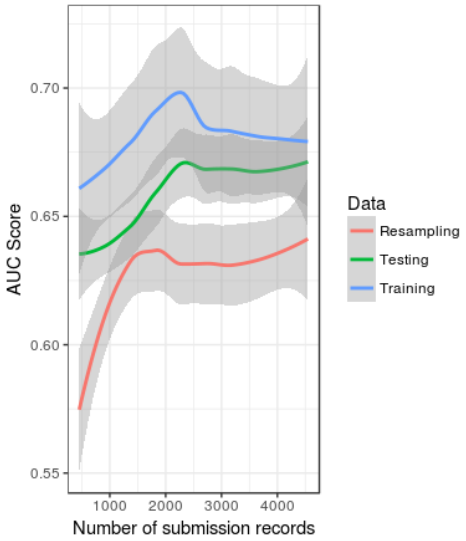

Figure 2: Training and testing Learning curves Random Forest

4.3.2 Parameter Tuning Using Cross Validation

In order to fine tune the parameters of our model, we use grid search in order to optimize number of trees, and size of subset of tried features. We then use cross validation with 10 iterations to train and test on different splits of the dataset which has two advantages: first it reduces the likelihood of overfitting as the model is trained on more data splits, second it returns the mean scores which gives a more accurate idea of the performance of the model. As figure 3 shows, we get the combination that gives us the best AUC score and which is used from now on by random forest model is mtry=2 and ntree=50.

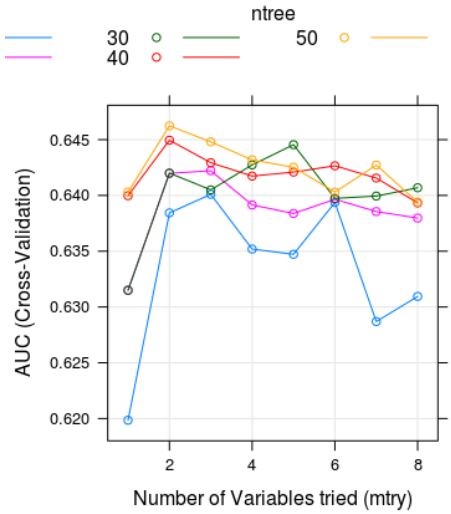

Figure 3: Parameter Tuning and Selection Random Forest

4.3.3 Confusion Matrices

Using the combination we got after fine tuning, we plot the confusion matrix to observe the disparities of the performance of the model with respect to the two different classes. By comparing the ratios of true positives and true negatives, we observe that the model does more or less similar performance predicting ”yes” and ”no” (doesn’t have a bias recognizing one class at the expense of the other. Figure 4 shows confusion matrix for training dataset using random forest which has slightly higher value for true positives than for true negatives and which does overall slightly less mistakes finding improvement than finding no improvement as the number of false positives is big compared to false negatives.

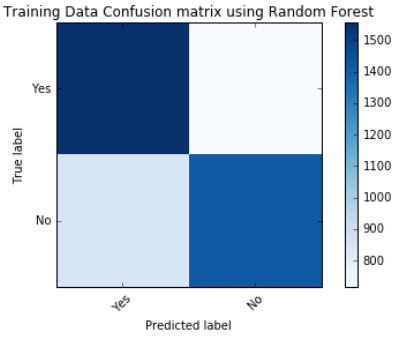

Figure 4: Confusion Matrix of predictions of training dataset

4.4 SVM

We applied Support Vector Machine to the dataset using different kernels: linear, radial, polynomial. In the following, we report the results we got using radial kernel (which gave us better results). After fine tuning, we got the best possible results with cost= 100 and gamma =2. We believe that lack of efficiency of this model is due to weak separability of the data no matter which kernel is used.

Table 1: Comparison of Different ML methods

Model /Caret Model	Model Parameters	Features Set	AUC (train)	AUC (validation)	AUC (testing)
Logistic Regression (LogitBoost)	nIter	7 features*	0.6021127	0.5805985	0.60466
RF (customized version)	mtry, ntree	7 features* , model kept only 2	**0.6725352**	**0.5995893**	**0.63933**
SVM (independent package)	kernel, cost, degree, gamma, coef.0, epsilon	8 features*	0.6272007	0.5785421	0.61099
K-NN (knn)	k	7 features*	0.6868398	0.5805955	0.58892
Neural Net(pcaNNet)	size, decay	7 features*	0.5944102	0.5785421	0.60358

4.5 Convolutional Neural Network

We tried different caret implementations of neural network: neural networks with pca, neural network with backpropagation and without. However, even after fine tuning parameters size=5 and decay=8, this algorithm doesn't give better results than random forest. The only promising venue here is using deep learning but training with bigger size of layers requires both time and computation power.

5 Performance Evaluation Method

In this project, rather than relying on accuracy which doesn't take into consideration the balance in the confusion matrix, we rely on AUC to evaluate the performance of our classifiers and predictive power of our features. Getting a score of 64% which is higher than 50% we can say that our best classifier does way better than random guessing.

6 Summary

In what follows the comparison of the performance of the different methods described above applied to this classification task. Table 1 shows that we got the best results for different splits of training (training and validation) and kaggle datasets using Random Forest and the lowest score was achieved using Logistic Regression.

We have tried other algorithms as well but their results are not better than the ones provided here. All in all, we can conclude based on the scores we got that there is medium correlation between the features selected and the ability to predict grade improvement. In more specific terms, the level of interaction between video and forum activities portrayed in feature engagement index, total video duration spent per each video and the number of submissions left all played an important role in predicting the student performance in the next submission. This is a main finding that can be used in designing and improving MOOCs as it suggests the following points:

- Video or forum activities alone are not as strong determinants as when combined together in an efficient way
- The more submission attempts the student can make the better his chances to get better grade which is not necessarily a predictor of his true understanding as he might for example not do any effort to watch videos or ask questions but can get better grade next time by trying other possibilities.
- The more time the student invests watching videos not just loading or skipping material the better his understanding of the material.

References

[1] Ren, Zhiyun, Huzefa Rangwala, and Aditya Johri. "Predicting Performance on MOOC Assessments using Multi-Regression Models." arXiv preprint arXiv:1605.02269 (2016).

[2] Gajos, Krzysztof Z., Juho Kim, Shang-Wen (Daniel) Li, Carrie J. Cai, and Robert C. Miller. 2014. Leveraging video interaction data and content analysis to improve video learning. In Proceedings of the CHI 2014 Learning Innovation at Scale workshop, Toronto, Canada, April 27, 2014.

[3] Mukala, Patrick, et al. "Learning Analytics on Coursera Event Data: A Process Mining Approach." (2015).

[4] Li, Nan, et al. "MOOC Video Interaction Patterns: What Do They Tell Us?." Design for Teaching and Learning in a Networked World. Springer International Publishing, 2015. 197-210.