Implementation of a Machine Learning-Based MOOC Recommender System Using Learner Motivation Prediction

https://doi.org/10.3991/ijep.v12i5.30523

Sara Assami[1(✉)], Najima Daoudi[2], Rachida Ajhoun[1]
[1]National School of Computer Science and Systems Analysis (ENSIAS), Mohammed V University in Rabat, Rabat, Morocco
[2]Information Sciences School (ESI), Rabat, Morocco
sara.assami@um5r.ac.ma

Abstract—The phenomenon of high dropout rates has been the concern of MOOC providers and educators since the emergence of this disruptive technology in online learning. This led to the focus on learner motivation studies from different aspects like demotivation signs detection, learning path personalization and course recommendation. Our paper aims to predict learner motivation for MOOCs to select the right MOOC for the right learner. Accordingly, we predict the motivation in an educational data mining approach by extracting and preprocessing learners' navigation traces on a MOOC platform, and building a Machine Learning model that predicts accurately a given learner's motivation for a MOOC. The comparison of the performance of four supervised learning algorithms resulted in the selection of the Random Forest classifier as the best modeling technique for motivation prediction with an accuracy of 95%. Afterward, we test the Machine Learning-based recommendation function for learners of the MOOC platform dataset to recommend the Top-10 MOOCs suitable for the target learner. Finally, further research on learner characteristics considered in recommender systems could enlarge the recommendation scope of MOOCs and maintain learner motivation.

Keywords—MOOC, recommender system, machine learning, learner motivation, learning analytics, course recommendation, classification algorithms, data preprocessing

1 Introduction

Massive Open Online Courses (MOOCs) refer to online courses where participants from all backgrounds can follow a course without subscription fees or qualifications. These online courses were developed in the higher education sphere [1], [2] and later proliferated for a lifelong learning experience. With the large numbers of MOOCs available for learners worldwide, MOOCs recommendation is a tool to guide learners in their learning quest. However, it's challenging to foresee the learners' interests with a range of learner profiles and a difficult understanding of elements impacting their motivation and engagement for a given MOOC [3].

In MOOC literature, the two concepts of motivation and engagement are intrinsically related. It is a learner's motivations (personal or professional reasons) towards subscribing to a MOOC that induces a positive or negative engagement towards this MOOC. Here, learner engagement in MOOCs "refers to student actions such as videos watched, quizzes answered and posts made to forums" [4]. Consequently, in a learning analytics approach, we need to analyze learners' engagement actions to define their motivation toward a MOOC.

Moreover, the learner engagement parameters' analysis will enable motivation prediction for a MOOC by a target learner. A prediction model will enable finding the relationship between an input and an output [5]. Although the model structure is set, its parameters are adjustable which allows the same model training on various data to deduce relationships for varying research problems [5].

As a first step of the learner motivation prediction, we use Educational Data Mining (EDM) techniques to preprocess a MOOC platform learning activities dataset.

In general, EDM is a technique used in learning analytics [6] that applies data mining techniques to analyze the learner's activities and get an overview of the learning context [7]. It was also used for online courses recommendation, like the research of [8] that developed an adaptive system for the recommendation of an English course learning sequence based on the learner profile. Ref. [8] collected data from an administered questionnaire to learners as input data for the Decision Tree algorithm to recommend the optimal learning sequence as an output. Much later, Ref. [9] performed preprocessing of data based on a distributed computation framework to recommend MOOCs by association rules data mining.

In educational data mining, Machine Learning (ML) algorithms like association rules and decision trees are not only used for course recommendation but are also used for MOOC dropout and demotivation prediction [10]. In a recent Systematic Literature Review of [11], twenty-seven research papers relied on ML techniques among 116 research works analyzed by [11] on MOOC recommender systems developed from 2013 to 2021.

Broadly, ML algorithms can be classified into two categories: supervised learning and unsupervised learning. However, new methods of a machine acquiring knowledge include reinforcement learning and transfer learning. This induced a large sphere of selected ML algorithms by researchers for MOOC recommendations such as k-means, K-Neighrest Neighbor, Apriori association algorithm, and decision tree [11].

Considering the related work identified by [11] as mentioned above and another systematic literature review of [12] on adaptive content recommenders in personalized learning environments from 2015 to 2020, the prediction model that we suggest in this paper is an ML-based prediction model for a MOOC personalized recommender. Indeed, the MOOC ML-based recommender "should decide two strategies: learner/learning object model and the recommendation technique" [12]. This paper aims to use the MOOC and learner data extraction from a MOOC platform dataset, and then predict the motivation of a learner for MOOCs based on historical learning data by resorting to EDM and ML algorithms. The paper's contribution relies upon the detailed description of preprocessing mechanisms of MOOCs data and the selection of adequate ML models for the classification problem of learner motivation prediction.

2 Materials and methodology

To generate a learner motivation prediction model, finding the right dataset that provides sufficient learner and MOOC features is essential for a researcher. In this sense, we use the Canvas Network open dataset that contains data from "Canvas Network open courses (running January 2014–September 2015)" [13]. The dataset contains more than 325.000 records where each record describes a learner activity in a MOOC from a list of 238 Canvas Network online courses [13]. It is the most recent dataset in terms of years of learner activity tracked (2014–2015) and offers 24 informed features with a balance between learner features and course features (compared to similar open datasets related to our research like the HarvardX-MITx dataset [14] or the Open University Learning Analytics dataset [15]).

In our previous work [16], we used the same dataset for learner profile enrichment which enabled a first data cleaning and feature selection of the data pre-processed on a Python notebook in "Google Colaboratory". The preprocessing steps in our previous work [16] included: irrelevant features removal, features renaming, datatype conversion, missing values detection, imputation of missing values, and data records sorting [16]. In this context, learner missing values are standard since it is due to survey questions unanswered by a learner at his subscription to a MOOC.

Figure 1 gives an overview of the obtained canvas dataset after the first preprocessing [16] that the authors use in this paper for the MOOC recommendation model. The dataset mainly provides three subsets of information: learner features, MOOC features, and learning activity features.

course_id	discipline	user_id	viewed	explored	grade	grade_reqs	...	course_start	course_end	last_event	nevents	nforum_posts	course_length
832945100	Interdisciplinary and Other	832400307	False	False	0.471	False	...	2014 Q1	2014 Q2	2015 Q3	524	4	124
832945100	Interdisciplinary and Other	832400582	False	False	0.029	False	...	2014 Q1	2014 Q2	2015 Q3	524	4	124
832945100	Interdisciplinary and Other	832401062	False	False	0.093	False	...	2014 Q1	2014 Q2	2015 Q3	524	4	124
832945100	Interdisciplinary and Other	832401808	False	False	0.628	False	...	2014 Q1	2014 Q2	2015 Q2	524	4	124
832945100	Interdisciplinary and Other	832402180	False	False	0.081	False	...	2014 Q1	2014 Q2	2015 Q3	524	4	124

Fig. 1. Canvas dataset preview after pre-processing and missing values imputation

Through the investigation of the historical data of the MOOC platform, we intend to predict whether a learner will be motivated or not to follow a MOOC. Practically, this implies a binary classification problem for which we need specific dataset processing and adequate ML algorithms selection for the prediction methodology.

2.1 Data pre-processing

Overall, The Canvas Network dataset preprocessing for MOOC recommendation purposes shows two major variable types: numerical variables (e.g.: user id and course length), and categorical variables (learner type, educational level…). For Dataset selection, feature engineering techniques are used to make data usable for ML algorithms of behavior prediction. The feature engineering for the Canvas Network dataset includes the following:

Variables removal: variables that are derived from the learner activity were removed (e.g.: number of forum posts, and number of events) since they are the result of a learner exploration of a MOOC, whereas we intend to analyze learner and MOOC features' values that contribute to one's motivation to follow a MOOC or not.

Dependent and independent variables definition: concerning our problem statement above, the target variable (e.g. dependent variable) from the Canvas Network dataset is the "explored" variable since it expresses the exploration of a given Canvas MOOC by a learner which shows one's motivation for a course. As for the independent variables, it comes to MOOC features (e.g.: course requisites, course start, and course end date) and the learner features from the survey data stored in each learner subscription to a course. The features list and data types are summarized in Table 1.

Categorical features transformation: ML algorithms usually require that each data input is represented as a numerical value to predict the probability of the response variable values. Thereby, we encode the categorical features using the one-hot encoder from the Sickit-learn library in Python to transform categorical features with more than two categories and convert the binary categorical features to the Boolean variable type (cf. Table 1 for feature values after categorical values numerical encoding).

Dataset sampling: As some ML models, like Random Forest, require a well-balanced representation of each predicted class in the dataset, we verify the number of rows with a "True" value and rows with a "False" value for the target variable "explored". The results show that the pre-processed dataset contains 138000 additional observations for MOOCs explored by learners compared to unexplored MOOCs. Therefore, we proceed to a data sampling with a random selection of 9711 observations for each predicted class category.

Table 1. Features 'selection for learner motivation prediction for a MOOC

Feature Type	Feature Name	Feature Data Type	Feature Values Encoding Type
MOOC features	course_reqs	Categorical (binary)	(bool)
	grade_reqs	Categorical (binary)	(bool)
	course_length	Numerical (discrete)	(Int64)
	course_start	Categorical (ordinal)	(Int64)
	course_end	Categorical (ordinal)	(Int64)
Learner features	encoded_primary_reason	Categorical (nominal)	(Int64)
	encoded_learner_type	Categorical (ordinal)	(Int64)
	expected_hours_week	Categorical (ordinal)	(Int64)
	encoded_education_level	Categorical (ordinal)	(Int64)
	explored	Categorical (binary)	(bool)

2.2 Methodology

In general, technology trends in MOOC recommender systems are heading towards the employment of ML algorithms [11]. Approaches applied to the learning context with a perspective of considering a wide set of variables for recommendation are Bayesian

networks, association rules, clustering, genetic algorithms, and semantics [17]. Since we already have a dataset that provides historical data about a learner's exploration of a course, supervised learning algorithms are the most adapted for our binary classification problem. Indeed, labeled input feature data is a prerequisite for supervised learning algorithms [18]. The latter use identified data properties to generate a model that accurately predicts labels for new data inputs [18].

Hence, we'll use the supervised learning models, specifically the classifiers adapted to our features data types (categorical and numerical independent features) to predict if a learner will have a true or false label for the probability of exploring a MOOC. Four ML models are used for this study to inspect the accuracy of learner motivation prediction, namely the Bayesian network, the Logistic Regression, the Support Vector Machine (SVM), and Random Forest models. It will be implemented in three stages.

First, we split the data into a training set of 70% of the Canvas dataset and 30% for the testing set. Second, we fit the model on the training set after importing each ML model function and defining its hyperparameters. Finally, we use the ML model to make predictions and measure its prediction's accuracy by using ML models' performance measures. Details of ML models' implementation and the evaluation of the model's performance will be given in the next section.

3 Machine learning models implementation

After the dataset uploading and preprocessing described in the previous section, it comes to the implementation of the selected ML algorithm to generate a classification model of learners' motivation depending on their learning activity history and their characteristics alongside the MOOC characteristics.

3.1 Naïve Bayes model

Broadly, a Bayesian function algorithm is used in supervised learning for classification. It mainly determines the probability of an event A given B when we already know the probability of A, the probability of B, and the probability of B given A as stated in the Bayes theorem equation [19]:

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

The Naïve Bayes classifier is based on the Bayes theorem and considers variables to be independent of each other as a hypothesis [19]. Consequently, we assume that all features are independent of each other [19], [20] and predict the likelihood of an event occurring (response variable value) based on evidence in our dataset (independent features values). In that, the Naïve Bayes classifier is known to be efficient and fast [21] and "one of the simplest classifiers" [19]. It has many application domains like the classification of news articles and emails, object recognition, weather prediction, etc.

Moreover, the Naïve Bayes model has three kernels: the multinomial, the Bernoulli, and the Gaussian kernel [18]. For our Naïve Bayes classifier implementation, we use

the Gaussian kernel since our independent features' values have a normal distribution (cf. Table 1: most features have ordinal values). Figure 2 shows the Naïve Bayes model implementation on Google Colaboratory.

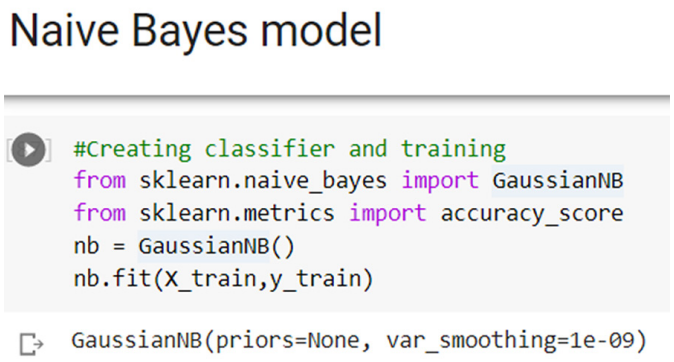

Fig. 2. Naïve Bayes model implementation

3.2 Logistic Regression model

A Logistic Regression algorithm is mainly used for binary classification to predict if something is true or false and is generally used "to estimate values for a categorical target variable" [18] that needs to be binary or ordinal [18]. Unlike the naïve classifier that ignores relationships among feature values, the Logistic Regression model is used "to model relationships between features in a dataset" [18] and enables us to test if a feature is significant for predicting the target value. Hence, it could help in confirming if a newly added learner or MOOC feature impacts the learner's motivation for a MOOC.

Still, the features need to be independent of each other for a **Logistic Regression** model but contrary to the gaussian Naïve Bayes model, they "are not required to have a normal distribution" [18], which gives a wider margin for learner profile enrichment with features of different value distribution.

For the implementation of the Logistic Regression model on our dataset, we specify the hyperparameter ***solver* = *'liblinear'*** which is used for small datasets such as our Canvas Network preprocessed dataset. Figure 3 illustrates the Logistic Regression model implementation.

Logistic Regression model

```
[39] # import model
     from sklearn.linear_model import LogisticRegression

[47] # STEP 2: train the model on the training set
     logreg = LogisticRegression(solver='liblinear')
     logreg.fit(X_train, y_train)

     LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                        intercept_scaling=1, l1_ratio=None, max_iter=100,
                        multi_class='auto', n_jobs=None, penalty='l2',
                        random_state=None, solver='liblinear', tol=0.0001, verbose=0,
                        warm_start=False)
```

Fig. 3. Logistic Regression model implementation

3.3 Support Vector Machine

Differing from the Naïve Bayes classifier, the SVM model is "a non-probabilistic binary classifier" [22] that starts with data in a low dimension and moves into a higher dimension to find the Support Vector Classifier (SVC) that separates the higher dimensional data into two groups by a hyperplane.

Moreover, SVM uses a kernel function to compute the relationship of a new observation with existing observations to find the SVC in higher dimensions. The basic kernel is *the linear classifier* "where SVM can classify data that can be separated linearly" [23]. If the number of features is large, the linear data mapping is sufficient to map data to a higher dimensional space [24]. However, in our Canvas dataset, we have a small number of features (9 features) which made us explore the nonlinear kernels that are the closest to real-world applications where features don't usually have a linear relation. In this sense, there are many popular nonlinear kernel functions like the polynomial, the Radial Basis Function (RBF), and the sigmoid kernel. For this study, we test three of the popular SVM kernels: linear, polynomial, and RBF kernel.

After the data is split into training and testing sets, we proceed with data scaling with the scale function from the Sickit-Learn Python library. The data scaling step aims to center the data before scaling to get a balance of each feature's data representation regardless of its numeric range for our SVM model implementation. The data scaling also provides the advantage of making calculations easier in SVM kernels. Kernel values are typically determined by the inner products of feature vectors like the linear kernel and the polynomial kernel where big attribute values may engender numerical issues [24].

Secondly, we implement the SVM model by importing the SVM function, creating the SVM classifier with a specification of the kernel, and giving the True value for the *probability parameter* that is essential to generate the ROC curve. The latter will be used in the next sections (cf. ML models performance) to measure the model performance and requires a probabilistic approach to prediction results which isn't the SVM characteristic by default.

Thirdly, we fit the model to the training set and make predictions for the testing set for the different SVM model kernels to compare their accuracy percentage and focus on the optimal kernel. Figure 4 shows the implementation of the linear kernel, the polynomial kernel, and the RBF kernel respectively.

Testing of the prediction accuracy results shows that the RBF kernel gives the highest accuracy score (92%) compared to the linear kernel (89%) and the polynomial kernel (91%) functions of SVM. Hence, we proceed to the parameter tuning of the RBF kernel to get the best accuracy score for the SVM model.

For the RBF kernel, the most influencing parameters are "C" and "gamma". The "C" parameter is for the regularization of SVM "classification of training examples against maximization of the decision function's margin" [25]. The gamma parameter defines the influence radius of dataset samples selected by the SVM model [25]. Therefore, we use the *grid search* technique to define the best (C, γ) pair for our SVM model. This technique establishes a finite number of alternative values for each parameter. Afterward, all conceivable combinations of these values are examined to achieve the optimal result [26].

Results of the selection of the best parameters from a list of finite numbers' are illustrated in Figure 5. We use these results to recreate the RBF SVM model (Gaussian model) with the newly defined parameters. The results demonstrate the increase in the accuracy percentage (94%) compared to the model generation before parameter tuning (92%).

RBF Kernel

```
[26] #Import svm model
     from sklearn import svm

     from sklearn.svm import SVC
     svm_gaussian = SVC(kernel='rbf', probability=True)
     svm_gaussian.fit(X_train, y_train)
```

```
[27] #predict test results
     y_pred = svm_gaussian.predict(X_test)

     #print accuracy
     print("Accuracy of Gaussian SVM in % :  " + str(accuracy_score(y_test,y_pred)*100))
```

Fig. 4. SVM kernels implementation

```
# Grid search
from sklearn.model_selection import GridSearchCV
parameters = {'C': [1, 10, 100],
          'gamma': [0.001, 0.01, 1]}
model = svm_gaussian
grid = GridSearchCV(estimator=model, param_grid=parameters, verbose = 0)
grid.fit(X_train, y_train)
print(grid)
# summarize the results of the grid search
print(grid.best_estimator_)
```

```
GridSearchCV(cv=None, error_score=nan,
             estimator=SVC(C=1.0, break_ties=False, cache_size=200,
                           class_weight=None, coef0=0.0,
                           decision_function_shape='ovr', degree=3,
                           gamma='scale', kernel='rbf', max_iter=-1,
                           probability=True, random_state=None, shrinking=True,
                           tol=0.001, verbose=False),
             iid='deprecated', n_jobs=None,
             param_grid={'C': [1, 10, 100], 'gamma': [0.001, 0.01, 1]},
             pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
             scoring=None, verbose=0)
SVC(C=100, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=True, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
```

Fig. 5. Grid search for SVM parameter tuning

3.4 Random Forest

Random Forest is an ensemble algorithm developed by Leo Breiman and used in supervised learning for classification and regression [19]. The Random Forest algorithm uses multiple independent decision trees classifiers with a double random drawing: the random draw of observations with a replacement on rows and the random draw of variables [19]. This process is called "bagging" [27] and its advantage compared to decision trees is that it constructs random trees and aggregates the tree's predictions to classify a new observation by using the voting method [28], this stops the error propagation risk of decision trees [18]. Though Random Forest requires extra computation [22] which makes it slower compared to decision trees [18], it is still efficient for its robustness [18], [28].

For Random Forest implementation (cf. Figure 6) to predict learner exploration of a course, we follow these steps:

- *Previous steps described for dataset preprocessing* (feature selection, categorical features transformation, data sampling, …) and Data split into two sets: training set (70%) and testing set (30%)
- *Data scaling* to get a balance of data representation by using the Min_Max_scaler function from Sickit-learn that scales all the data features in the default range [0, 1].
- *Model fitting:* we fit the Random Forest classifier to the data with the definition of the *n_estimators* parameter which defines the number of trees to generate by the model. As we have few features and a small dataset, we define a small number of 10 for this parameter.

- *Accuracy calculation* for the training set and the testing set: the computation of the accuracy for each split of data is done to verify if the model isn't overfitted to the training set since the overfitting problem is very common in ML models generally and Random Forests specifically [28].

```
# Data scaling
min_max_scaler = preprocessing.MinMaxScaler()
X_train = min_max_scaler.fit_transform(X_train)
X_test= min_max_scaler.fit_transform(X_test)
```

```
[105] #Import Random Forest Model
      from sklearn.ensemble import RandomForestClassifier

      #Create a Gaussian Classifier
      clf=RandomForestClassifier(n_estimators=10)

      #Train the model using the training sets y_pred=clf.predict(X_test)
      clf.fit(X_train,y_train)

      # make predictions
      y_pred=clf.predict(X_test)
```

```
[106] # compute accuracy of predictions on training set
      clf.score(X_train,y_train)

      0.9524825303420376
```

```
[107] # compute accuracy of predictions on testing set
      clf.score(X_test,y_test)

      0.94371031405526
```

Fig. 6. Random Forest model implementation

4 Machine learning models performance and prediction evaluation

Since our prediction model is based on a classifier, we can use many prediction quality metrics to evaluate the classification model like the prediction accuracy measures, the confusion matrix, the ROC curve, and the Area Under the Curve (AUC). In addition, accuracy, precision, recall, and F-measure are the most used evaluation metrics of MOOC recommender systems in general [11].

4.1 The prediction accuracy measures

Above all, accuracy is the percentage of successful predictions from all the predictions [21]. In classification, the accuracy score is the percentage of correctly predicted labels in the true subset labels. Nonetheless, other measures investigate deeper the predictions' correctness and error metrics, especially for an ML model intended for the recommendation.

First of all, the precision measure calculates the proportion of pertinent recommendations from the given recommendations. Therefore, it focuses on the results of a recommender and is given by [29]:

$$Precision = \frac{T_p}{T_p + F_p}$$

Where:

T_p: True positive, e.g. the number of pertinent recommendations

F_p: False positive, e.g. the number of false recommendations

Secondly, the recall measure calculates the capacity of a recommender to give pertinent recommendations. It focuses on the pertinence of a recommender compared to what it should recommend. Hence, it is given by [29]:

$$Recall = \frac{T_p}{T_p + F_n}$$

Where:

F_n: False-negative, e.g.: pertinent recommendations that weren't given by the RS.

Subsequently, the F-measure comes to combine both precision and recall into a single measure [30]. It is used for recommender systems and computes the harmonic mean of precision and recall [30]. It is also the standard measure to evaluate a binary classifier [20] like our prediction classifier for learner exploration of a MOOC. Its formula is:

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

4.2 The confusion matrix

The utility of a confusion matrix is that it compares observed data and predicted data for each class and shows the misclassification rate for each class to predict by the binary classifier. The confusion matrix is a two-by-two matrix that shows how many points in a testing data were assigned to a category compared to where they should be assigned [20].

It becomes an essential tool to evaluate the classifier performance for each class [31], especially when the model accuracy rate is very high but the classification's error rate for a class is much higher than the error rate for the other class.

4.3 The ROC curve and the Area Under the Curve

The Receiver Operating Characteristic (ROC) curve is constructed by using two performance indicators [19]:

- the specificity β: the False Positive Rate (FPR) with 1-β for the x-axis and which represents the number of negative examples that are predicted correctly;
- the sensitivity α: the True Positive Rate (TPR) on the y-axis, which represents "the fraction of all hits that are correctly classified as hits" [20].

The specificity and sensitivity indicators are in the range [0, 1] and are measured by varying the thresholds of the confusion matrix each time to get a curve point $(1 - \beta, \alpha)$.

Hence, the ROC curve enables a common ground for different ML models comparison. This comparison could be done globally regardless of the decision threshold by considering the Area Under the Curve (AUC) [19] a common measure used for depicting the accuracy of classifiers from ROC curves based on the same data. Indeed, the larger the AUC is, the better is the model, whereas a model with a ROC curve under an AUC = 0.5 has a problem of predicting the Positive as Negative and the Negative as Positive.

4.4 The ML models' performance comparison

After the ML models implementation, we verify by using the confusion matrix for each of the 4 models that the algorithms' accuracy prediction for each class doesn't have a significant gap with the prediction accuracy for its opposite class. In general, the main metrics used for a classification problem are the F1-measure and the ROC Curve [19]. The F1-measure "will be 1.0 for a perfect classifier and 0.0 in the worst case" [20] and the ROC curve could be represented by its AUC. Nonetheless, the accuracy score is a common measure for all ML models and is worth using in the performance comparison.

In our ML models' prediction results (cf. Figure 7), the accuracy is between 89% and 95%. Therefore, the chosen ML models have a good performance on the dataset. What's more is that for classification and recommendation purposes, the F1-measure high values obtained for all 4 models with a similar range of the accuracy scores [0.89 – 0.95] also assure the suitability of all models for our classification problem and could be used to predict interchangeably the learner exploration of a course or not.

Nonetheless, the main cutting value for the designation of the most performing classifier is the AUC value, which also had values for all four models near the optimum value 1, but the Random Forest model attained the largest AUC of 0.98 with a very fast execution time of 0.059 seconds compared to the SVM model that had a close AUC of 0.95 but took more than 46 seconds to fit the model to the dataset.

In consequence, the Random Forest is the best ML model compared to the SVM, the Logistic Regression, and the Naïve Bayes classifiers that we could use for learner motivation prediction.

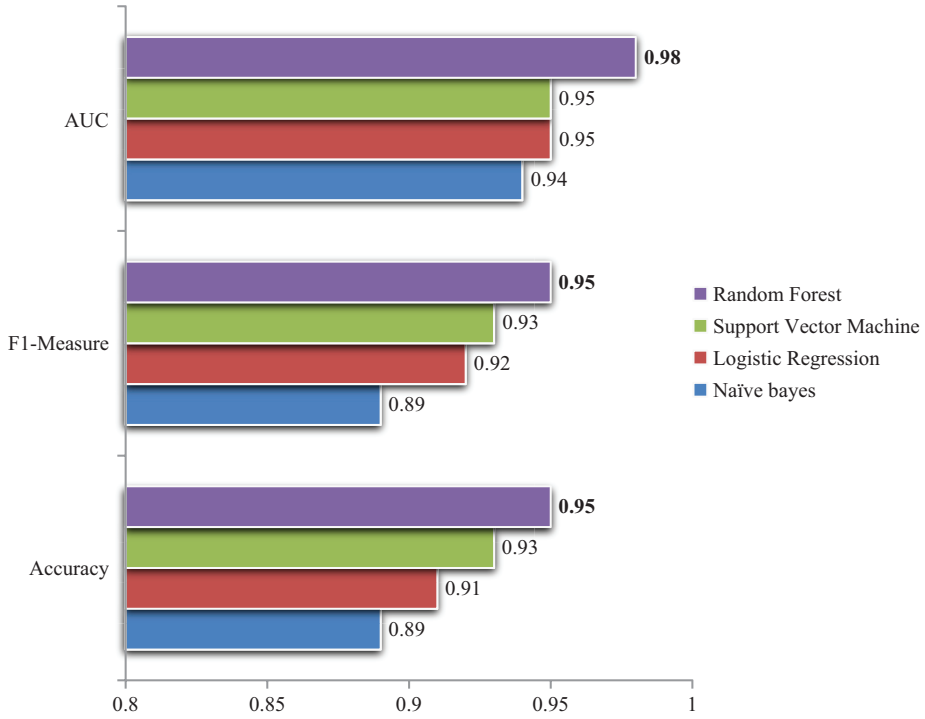

Fig. 7. Comparison of classifiers' performance for prediction of learner motivation

5 Machine Learning-based MOOC recommender

The ML-based recommender that we suggest takes as input only the learner ID and uses his historical data to recommend adequate MOOCs. This recommendation function excludes the MOOCs with a null label for the motivation prediction of the target learner. It concretizes the approaches of many research papers and platforms of adopting a filtering approach based on learner characteristics and MOOC metadata.

To illustrate the MOOC recommender system functioning, Figure 8 displays an example of a target learner profile: ready to dedicate a maximum of 4 hours of work per week to follow a MOOC but considers themselves as an active learner and has personal motivations for topics of interest. After selecting the target learner for the RS, we test the ML-based MOOC recommender function that uses the Random Forest previously generated model to predict which MOOCs will be explored by the target learner and generate the top-10 recommended courses list (cf. Figure 9).

	user_id	primary_reason	learner_type	expected_hours_week	education_level
0	832400307	I enjoy learning about topics that interest me	Active	Between 2 and 4 hours	Master's Degree (or equivalent)

Fig. 8. Target learner for ML-based MOOC recommender

```
[190] ML_recommender(learner_id)
```

	course_id	explore	discipline_text	grade_reqs	course_reqs	course_length
21	832945145	True	humanities	True	True	60
103	832945515	True	mathematics statistics	True	True	35
121	832945565	True	education	True	True	77
130	832945591	True	professions applied sciences	True	True	365
183	832960448	True	interdisciplinary	True	True	122
188	832960714	True	business management	True	True	365
193	832960719	True	professions applied sciences	True	True	35
195	832960721	True	professions applied sciences	True	True	363
200	832960758	True	humanities	True	True	42
210	832960903	True	interdisciplinary	True	True	47

Fig. 9. Machine Learning-based MOOC recommender result

As Figure 9 demonstrates, the recommended MOOCs have varying disciplines and course lengths but all share the same course and grade requisites. This infers that the recommender system predicts that the target learner will be more motivated for courses with pre-requisites and a learning level evaluation.

Consequently, the suggested MOOCs by the recommender satisfy the personal curiosity of the target learner for following MOOCs since they are about different domains and enable the learner to choose a MOOC with a course length that is adequate with a tight schedule (4 hours per week amount of work).

Furthermore, we also use the features importance technique on the Random Forest implemented model to explore the most influential features on learner behavior prediction (cf. Table 1 for list of features). The bar chart in Figure 10 summarizes the feature importance results for our dataset.

Figure 10 shows that the features that influenced greatly the motivation of the Canvas Network learners are mainly the MOOC characteristics (eg.: grade and course requisites, course length and course start and end date). This is predictable as the dataset of the Canvas Network provides precise information about MOOCs characteristics, whereas the learner features values had an important amount of missing values (48.4% of dataset records with missing values for all learner features). Consequently, the algorithm learned better from MOOC feature values than synthetic imputed values of learner features missing data.

For this reason, a greater focus on learner data collection is necessary to get learner feature values (eg.: learner type, education level, primary reason and expected hours per week) and balance their importance compared to MOOCs features for the MOOC recommendation criteria.

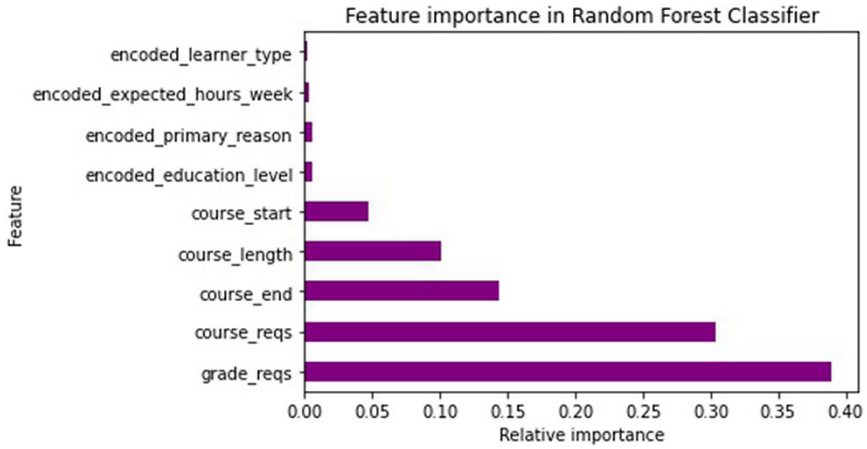

Fig. 10. Learner and MOOC features' importance from the Random Forest classifier

6 Conclusion and future work

At present, learner motivation is a key element for MOOC recommendation and increasing the MOOC completion rates with the proliferation of MOOCs and the increasing dropout rates [32], [33]. Upon the interdependence of learner motivation and achievement in MOOCs [33], we studied learner and MOOC data on MOOC platforms and selected an adequate dataset for the implementation of a MOOC recommender. The latter aimed to explore the historical data on the MOOC platform Canvas Network [13] and pre-process it to enable the recommendation function's learning about interesting MOOCs to learners and recommend the top-10 most adequate online courses for a given learner.

Subsequently, we selected the ML algorithms suitable for learner profile classification into one of the two categories: is likely to explore or not likely to explore a MOOC. The four implemented ML classifiers: Naïve Bayes, Logistic Regression, SVM, and Random Forest gave similar performance percentages but the Random Forest-based model gave the highest accuracy rate (95%).

Therefore, the conception of the MOOC recommender ML-based system used the Random Forest model for learner motivation prediction and relied on both learner and MOOC characteristics. The testing of the ML-based recommender concretizes the useful inclusion of ML in MOOCs list filtering since it provides a personalized list of MOOC items to learners.

Furthermore, we concluded from the feature importance algorithm that MOOC features have a bigger influence, compared to the personal characteristics of learners, on the exploration or not of a given MOOC.

However, using ML to recommend MOOCs similar to MOOCs previously explored by learners is a content-based recommendation since it is based solely on MOOC features and doesn't take into account the individual characteristics of a learner.

Additionally, the Canvas dataset learner features don't include the knowledge domains of interest for a learner or the sought skills to develop through MOOCs. Even if such data is available, ML algorithms don't enable the semantic matching of learner domains of interest with MOOCs disciplines.

For these research limitations, we will be exploring in future works the hybridization of ML techniques with the ontology-based recommendation approach as suggested by [34] to upgrade the variety of learner features used for recommendation in adherence to the MOOC recommendation criteria. In this sense, educational resources management systems use ontologies to share information and annotate semantically these resources [35]. The semantic annotations will enable matching MOOC content knowledge with the sought knowledge by a target learner. Consequently, it'll expand the potential learner features used for MOOC recommendation.

7 References

[1] B. Kieslinger, J. Tschank, T. Schaefer, and C. M. Fabian, "Working in Increasing Isolation? How an International MOOC for Career Professionals Supports Peer Learning across Distance," *Int. J. Adv. Corp. Learn.*, vol. 11, no. 1, p. 23, 2018, https://doi.org/10.3991/ijac.v11i1.9117

[2] S. Vorbach, E. Maria Poandl, and I. Korajman, "Digital Entrepreneurship Education: The Role of MOOCs," *Int. J. Eng. Pedagog.*, vol. 9, no. 3, pp. 99–111, 2019, https://doi.org/10.3991/ijep.v9i3.10149

[3] S. Assami, N. Daoudi, and R. Ajhoun, "A Semantic Recommendation System for Learning Personalization in Massive Open Online Courses," *Int. J. Recent Contrib. Eng. Sci. IT*, vol. 8, no. 1, pp. 71–80, 2020, [Online]. Available: https://doi.org/10.3991/ijes.v8i1.14229

[4] J. Sinclair and S. Kalvala, "Student Engagement in Massive Open Online Courses," *Int. J. Learn. Technol.*, vol. 11, no. 3, pp. 218–237, 2016, https://doi.org/10.1504/IJLT.2016.079035

[5] E. Alpaydin, *Machine Learning*. The MIT Press, 2016.

[6] W. M. A. F. Wan Hamzah, I. Ismail, M. K. Yusof, S. I. M. Saany, and A. Yacob, "Using Learning Analytics to Explore Responses from Student Conversations with Chatbot for Education," *Int. J. Eng. Pedagog.*, vol. 11, no. 6, pp. 70–84, 2021, https://doi.org/10.3991/ijep.v11i6.23475

[7] N. Manouselis, H. Drachsler, K. Verbert, and E. Duval, "Introduction and Background BT – Recommender Systems for Learning," in *SpringerBriefs in Electrical and Computer Engineering*, N. Manouselis, H. Drachsler, K. Verbert, and E. Duval, Eds. New York, NY: Springer New York, 2013, pp. 1–20, https://doi.org/10.1007/978-1-4614-4361-2_1

[8] Y. huei Wang, M. H. Tseng, and H. C. Liao, "Data Mining for Adaptive Learning Sequence in English Language Instruction," *Expert Syst. Appl.*, vol. 36, pp. 7681–7686, 2009, https://doi.org/10.1016/j.eswa.2008.09.008

[9] H. Zhang, T. Huang, Z. Lv, S. Y. Liu, and Z. Zhou, "MCRS: A Course Recommendation System for MOOCs," *Multimed. Tools Appl.*, pp. 1–19, 2017, https://doi.org/10.1007/s11042-017-4620-2

[10] F. Dalipi, A. S. Imran, and Z. Kastrati, "MOOC Dropout Prediction using Machine Learning Techniques: Review and Research Challenges," in *2018 IEEE Global Engineering Education Conference (EDUCON)*, 2018, pp. 1007–1014, https://doi.org/10.1109/EDUCON.2018.8363340

[11] I. Uddin, A. S. Imran, K. Muhammad, N. Fayyaz, and M. Sajjad, "A Systematic Mapping Review on MOOC Recommender Systems," *IEEE Access*, vol. 9, pp. 118379–118405, 2021, https://doi.org/10.1109/ACCESS.2021.3101039
[12] N. S. Raj and V. G. Renumol, "A Systematic Literature Review on Adaptive Content Recommenders in Personalized Learning Environments from 2015 to 2020," *J. Comput. Educ.*, 2021, https://doi.org/10.1007/s40692-021-00199-4
[13] Canvas Network, "Canvas_Network_Person-Course_Documentation.pdf," *Canvas Network Person-Course (1/2014 – 9/2015) De-Identified Open Dataset*, 2016, https://doi.org/10.7910/DVN/1XORAL/RLHZYZ (accessed Sep. 09, 2021).
[14] HarvardX, "HarvardX Person-Course Academic Year 2013 De-Identified Dataset, version 3.0." Harvard Dataverse, 2014, https://doi.org/10.7910/DVN/26147
[15] C. Zhenghao, B. Alcorn, G. Christensen, N. Eriksson, D. Koller, and E. J. Emanuel, "Who's Benefiting from MOOCs, and Why Who' s Benefiting from MOOCs, and Why," *Harvard Business Review*, pp. 1–9, 2015.
[16] S. Assami, N. Daoudi, and R. Ajhoun, "Learner Profile Enrichment and Semantic Modeling of Learning Actors for MOOC Recommendation," in *International Conference on Smart Systems and Data Science 2021 (ICSSD'21)*, 2021, p. In press.
[17] P. Montuschi, F. Lamberti, V. Gatteschi, and C. Demartini, "A Semantic Recommender System for Adaptive Learning," *IT Prof.*, no. September/October, pp. 50–58, 2015, https://doi.org/10.1109/MITP.2015.75
[18] L. Pierson, *Data Science for Dummies.*, 2nd ed. Wiley, 2017.
[19] E. Biernat and M. Lutz, *Data Science: fondamentaux et études de cas*. Eyrolles, 2015.
[20] F. Cady, *The Data Science Handbook*. Wiley, 2017, https://doi.org/10.1002/9781119092919
[21] Henri Laude and Eva Laude, *Data Scientist et langage R*, 2nd ed. Editions ENI, 2018.
[22] S. Liu, L. Zhang, and Z. Yan, "Predict Pairwise Trust Based on Machine Learning in Online Social Networks: A Survey," *IEEE Access*, vol. 6, no. September, pp. 51297–51318, 2018, https://doi.org/10.1109/ACCESS.2018.2869699
[23] N. A. Utami, W. Maharani, and I. Atastina, "Personality Classification of Facebook Users According to Big Five Personality Using SVM (Support Vector Machine) Method," *Procedia Comput. Sci.*, vol. 179, no. 2020, pp. 177–184, 2021, https://doi.org/10.1016/j.procs.2020.12.023
[24] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A Practical Guide to Support Vector Classification." National Taiwan University, Taipei, pp. 1–16, 2016, Accessed: Sep. 11, 2021. [Online]. Available: https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf
[25] Scikit-Learn Developers, "Documentation of Scikit-Learn 0.15," 2014, https://scikit-learn.org/0.15/documentation.html# (accessed Sep. 11, 2021).
[26] T. Eitrich and B. Lang, "Efficient Optimization of Support Vector Machine Learning Parameters for Unbalanced Datasets," *J. Comput. Appl. Math.*, vol. 196, pp. 425–436, 2006, https://doi.org/10.1016/j.cam.2005.09.009
[27] F. Di Troia, "Machine Learning Classification for Advanced Malware Detection," Kingston University, London, 2021.
[28] O. A. M. Rado, "Contributions to Evaluation of Machine Learning Models. Applicability Domain of Classification Models," University of Bradford, 2019.
[29] R. Sharma and R. Singh, "Evolution of Recommender Systems from Ancient Times to Modern Era: A Survey," *Indian J. Sci. Technol.*, vol. 9, no. 20, pp. 1–12, 2016, https://doi.org/10.17485/ijst/2016/v9i20/88005
[30] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds., *Recommender Systems Handbook*. London: Springer, 2011, https://doi.org/10.1007/978-0-387-85820-3

[31] I. Khan, A. R. Ahmad, N. Jabeur, and M. N. Mahdi, "Machine Learning Prediction and Recommendation Framework to Support Introductory Programming Course," *Int. J. Emerg. Technol. Learn.*, vol. 16, no. 17, pp. 42–59, 2021, https://doi.org/10.3991/ijet.v16i17.18995

[32] D. F. O. Onah and J. E. Sinclair, "Assessing Self-Regulation of Learning Dimensions in a Stand-Alone MOOC Platform," *Int. J. Eng. Pedagog.*, vol. 7, no. 2, p. 4, 2017, https://doi.org/10.3991/ijep.v7i2.6511

[33] C. Reparaz, M. Aznárez-Sanado, and G. Mendoza, "Self-Regulation of Learning and MOOC Retention," *Comput. Human Behav.*, vol. 111, no. January, 2020, https://doi.org/10.1016/j.chb.2020.106423

[34] C. Obeid, I. Lahoud, H. El Khoury, and P.-A. Champin, "Ontology-Based Recommender System in Higher Education," in *Companion Proceedings of the Web Conference 2018*, 2018, pp. 1031–1034, https://doi.org/10.1145/3184558.3191533

[35] C. Abderrahman, O. Aziz, and K. Mohamed, "Semantic Annotation of Resources of Distance Learning Based Intelligent Agents," *Int. J. Eng. Pedagog.*, vol. 4, no. 1, p. 69, 2014, https://doi.org/10.3991/ijep.v4i1.2845

8 Authors

Sara Assami received her Ph.D. degree in computer science from the National Superior School of Computer Science and Systems Analysis (ENSIAS), Mohammed Vth University in Rabat, Morocco. The Ph.D. research works developed the conception of personalized recommender systems for MOOCs for a learner-centered approach. Her research interests include: online personalized learning, semantic web technologies, Natural Language Processing and Educational Data Mining.

Najima Daoudi is a Professor at the School of Information Sciences, Rabat, Morocco. She is an Engineer of the National Institute of Statistics and Applied Economics and has a Ph.D. in Computer Science from ENSIAS. She has produced several articles in E-learning, M-learning and Ontology development since 2005. She was chair of the international conference ICSSD'21. (Email: ndaoudi@esi.ac.ma)

Rachida Ajhoun is a computer science Professor of higher education at the Mohammed V University in Rabat, specifically at the ENSIAS, Rabat, Morocco. She obtained her Ph.D. in 2001. Her main research field is the improvement of the online training techniques (E-Learning and M-learning). She is the author of the GCM Model (Generic Course Model) for the production of adaptative courses on the Internet. (Email: rachida.ajhoun@ensias.um5.ac.ma).

Article submitted 2022-03-01. Resubmitted 2022-06-20. Final acceptance 2022-06-20. Final version published as submitted by the authors.