Progressive Prediction of Student Performance in College Programs

Jie Xu,[*] Yuli Han,[†] Daniel Marcu,[**] Mihaela van der Schaar[†]
[*]University of Miami, Coral Gables, FL 33124
[†]University of California Los Angeles, Los Angeles, CA 90095
[**]Information Sciences Institute, University of Southern California, Marina del Ray, CA 90292

Abstract

Accurately predicting students' future performance based on their tracked academic records in college programs is crucial for effectively carrying out necessary pedagogical interventions to ensure students' on-time graduation. Although there is a rich literature on predicting student performance in solving problems and studying courses using data-driven approaches, predicting student performance in completing college programs is much less studied and faces new challenges, mainly due to the diversity of courses selected by students and the requirement of continuous tracking and incorporation of students' evolving progresses. In this paper, we develop a novel algorithm that enables progressive prediction of students' performance by adapting ensemble learning techniques and utilizing education-specific domain knowledge. We prove its prediction performance guarantee and show its performance improvement against benchmark algorithms on a real-world student dataset from UCLA .

Introduction

Making college affordable has a significant impact on ensuring the nation's economic prosperity and is a central focus of the government when making education policies (The White House 2016). Yet student loan debt in the United States has blown past the trillion-dollar mark, exceeding American's combined credit card and auto load debts (Complete College America 2014). As the cost in college education (tuition, fees and living expenses) has skyrocketed over the past few decades, prolonged graduation time has become a crucial contributing factor to the ever-growing student loan debt. In fact, recent studies show that only 50 of the more than 580 public four-year institutions in the United States have on-time graduation rates at or above 50 percent for their full-time students (Complete College America 2014).

To make college more affordable, it is thus crucial to ensure that many more students graduate on time through early interventions on students whose performance will be unlikely to meet the graduation criteria of the college program on time. A critical step towards the effective intervention is to build a system that can continuously keep track of students' academic performance and accurately predict their future performance, such as when they will graduate and their potential final GPAs, given the current progress. Although predicting student performance has been extensively studied in the literature, it was primarily studied in the contexts of solving problems in Intelligent Tutoring Systems (ITSs) (Cen, Koedinger, and Junker 2006)(Feng, Heffernan, and Koedinger 2009)(Yu et al. 2010)(Pardos and Heffernan 2010) , or completing courses in traditional classroom settings or in Massive Open Online Courses (MOOC) platforms (Meier et al. 2015)(Brinton and Chiang 2015). However, predicting student performance at the college program level is significantly different and faces new challenges.

First, students can differ tremendously in terms of pre-college traits as well as selected courses and the sequence in which they take the selected courses, especially since more and more institutions offer open curricula where students can take courses without concern for any requirements except those in their chosen concentrations (majors). In contrast, solving problems in ITSs often follow routine steps which are the same for all students (KDD Cup 2010). Similarly, predicting student performance in courses are often based on in-course assessments which are designed to be the same for all students (Meier et al. 2015).

Second, predicting student performance in a college program is not a one-time task; rather, it requires continuous tracking and updating as the student finishes new courses over time. An important consideration in this regard is that the prediction needs to be made based on not only the most recent snapshot of the student accomplishments but also the evolution of the student progress, which may contain valuable information of the student's learning style, habits and goals for making accurate predictions. However, the complexity can easily explode, which grows exponentially in the number of academic terms (quarters, semesters).

Third, whereas problems and courses often remain the same over a relatively long period of time, changes in college programs can be much more frequent due to the addition and removal of courses, and the adjustment of course requirement and availability in different quarters. As a result, prediction algorithms developed offline solely relying on a logged dataset can be outdated and ineffective in making accurate predictions. It is much desired that the predictions algorithm can be easily updated using new student data

Y. Han and M. van der Schaar's research is supported by NSF ECCS 1407712. This research was done when Y. Han was a visiting summer student at UCLA.

as it is being applied.

In this paper, we aim to address all aforementioned challenges by proposing a novel algorithm for predicting student performance in a college program. The algorithm adopts a bilayered structure comprising a base layer and an ensemble layer. In the base layer, base predictors make local predictions given the snapshot of the student academic states. They are trained offline on logged student data for every quarter utilizing the educational domain knowledge. In the ensemble layer, in each quarter, an ensemble predictor makes a final prediction by synthesizing the local prediction of the corresponding base predictor and the final prediction of the previous quarter ensemble predictor. The ensemble predictors are trained offline and updated online using new student data based on the Prediction with Expert Advice (PWEA) algorithm (Cesa-Bianchi and Lugosi 2006). We prove worst-case performance guarantee for our algorithm and evaluate its performance using a real-world educational dataset from a major university in the United States.

Related Work

Machine learning for education has gained much attention in recent years. A substantial amount of literature focuses on predicting student performance in solving problems or completing courses. Many machine learning techniques, such as decision trees (Marquez-Vera, Romero, and Ventura 2010), artificial neural networks (Wang and Liao 2011), matrix factorization (Thai-Nghe et al. 2011), collaborative filters (Toscher and Jahrer 2010) and probabilistic graphical models (Bekele and Menzel 2005)(Pardos and Heffernan 2010), have been applied to develop prediction algorithms. Most of this work ignores the temporal/sequential effect that students improve their knowledge over time and treats the prediction as a one-time task. To take the temporal/sequential effect into account, a three-mode tensor factorization (on student/problem/time) technique was developed for predicting student performance in solving problems in ITSs (Thai-Nghe, Horváth, and Schmidt-Thieme 2010) and a similarity-based algorithm was proposed to issue predictions of student grades in courses only when a certain confidence level is reached (Meier et al. 2016). However, due to the aforementioned substantial differences of predicting student performance in college programs, these methods are not applicable in our setting.

There is also a rich literature on recommending relevant courses or problems to students based on their associated knowledge level, learning styles, and feedbacks (Lee and Brunskill 2012)(Mandel et al. 2014)(Brunskill and Russell 2010). Course sequence recommendation, which considers the specific course constraints, was studied in (Xu, Xing, and van der Schaar 2016). To utilize logged data for course sequence recommendations and curriculum design, an off-policy estimator was developed to estimate how an unobserved policy performs given an observed policy (Hoiles and van der Schaar 2016). A rank aggregation framework is adapted for the discovery of optimal course sequences at the university level (Cucuringu et al. 2016). However, whereas this literature aims to recommend courses/course sequences based on student background and past performance, the purpose of the current work is to predict future performance based on student background and past performance for a given curriculum.

Our algorithm uses the ensemble learning technique, in particular, the Exponentially Weighted Average Forecaster (EWAF) (Cesa-Bianchi and Lugosi 2006) as a building block to enable progressive prediction of student performance and online updating of the predictor as new student data is received. The major difference from the conventional EWAF algorithm is that an ensemble predictor has access to only one base predictor (expert) and the previous quarter ensemble predictor, whose output summarizes the outputs of all previous quarter base predictors whereas the conventional EWAF algorithm has access to all experts directly. To our best knowledge, this is a novel architecture for designing predictors for progressively expanding input spaces, which significantly reduces design and implementation complexity and easily scales with the number of academic terms. In this setting, we prove that each ensemble predictor still performs asymptotically no worse than the best base predictor in hindsight among all previous quarter base predictors in the worst case, thereby providing strong performance guarantee.

System Model

Consider a college program in which students have to complete courses specified by curriculum $\mathcal{C}$ to graduate. A student is said to graduate on time if he/she completes the curriculum within T quarters where T is a predefined number. Students may start the program with different background such as different high school GPAs and SAT scores. Denote student i's background by $\theta_i \in \Theta$ where Θ is the space that includes all possible background. A student i can take any course(s) in any quarter subject to course availability and prerequisite requirements. Some of the courses may belong to the curriculum $\mathcal{C}$ and the rest may be extra-curricular courses selected according to the student's own interest. Once having completed a course, student i receives a grade for the course and earns the credits. Let $s_i^t \in \mathcal{S}^t$ denote student i's academic state at the beginning of quarter $t = \{1, 2, ...\}$, which includes information of what courses have been completed, the earned credits and the corresponding grades. Notice that (1) different students may take courses in different orders, (2) a student may take the same course multiple times due to failure and (3) students have different interests so they take different extra-curricular courses. As a result, the completed courses at the beginning of the same quarter t can differ significantly across students so the academic state space $\mathcal{S}^t$ can be extremely large.

Given the student's background and evolving academic state, at the beginning of each quarter t, the objective of our system is to predict when the student is able to graduate (or whether the student will graduate on time). Specifically, at the beginning of each quarter t, we construct a predictor $f^t : \Theta \times \mathcal{S}^1 \times ... \times \mathcal{S}^t \to \mathcal{Y}$ where $\mathcal{Y}$ is the space of all possible graduation time. Denote $\hat{y}_i^t = f^t(\theta_i, s_i^1, ..., s_i^t)$ as the predicted graduation time given the academic states up to quarter t and y_i as the actual graduation time of student i. We emphasize that the predictor is using not only a single

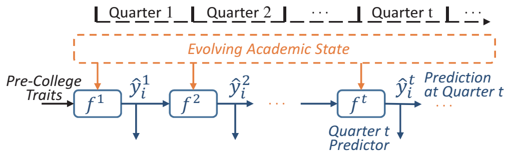

Figure 1: System architecture. Prediction results of the previous quarter can be utilized in the prediction of the current quarter.

academic state s_i^t but all the academic states in the previous quarters $s_i^1, ..., s_i^{t-1}$, thereby incorporating the evolution of student's performance. Even if two students have the same academic states at the beginning of quarter t, they may have taken completely different routes to reach this state. Taking into account such differences is necessary in the prediction of the student's future performance. However, the input space of the predictor grows exponentially in the number of quarters, thereby making the predictor construction even more challenging.

Proposed Method

To address the aforementioned challenges, we propose a novel framework for designing predictors for progressively expanding input spaces. The principle idea is that since the input of predictor f^t for quarter t is a superset of the input of predictor f^{t-1} for quarter $t-1$, f^t can capitalize on the prediction output $\hat{y}_i^{t-1}$ of f^{t-1} to update the prediction by incorporating the incrementally new information s_i^t. This significantly reduces the complexity of constructing f^t and makes the prediction algorithm scalable. This idea and the envisioned system architecture is illustrated in Figure 1.

Our approach to enable such progressive predictions is based on the ensemble learning techniques. The proposed architecture consists of two layers – a base layer and an ensemble layer.

- In the base layer, we construct a set of base predictors, each for one quarter. Specifically, a base predictor for quarter t is a function $h^t : \Theta \times \mathcal{S}^t \to \mathcal{Y}$ which utilizes only the academic state at the beginning of quarter t but not that of previous quarters. Let $z_i^t = h^t(\theta_i, s_i^t)$ denote the prediction result of h^t for student i.
- In the ensemble layer, we construct a set of ensemble predictors, each for one quarter. Each ensemble predictor f^t for quarter t synthesizes the output of the previous ensemble predictor $\hat{y}_i^t$ and that of the base predictor z_i^t for student i and makes a final prediction based on $\hat{y}_i^t$ and z_i^t.

A system block diagram of the proposed bilayered architecture for the quarter t ensemble learning is illustrated in Figure 2. In the next sections, we describe how to construct the base predictors and the ensemble predictors.

Learning Quarter-wise Base Predictors

In this section, we describe how to learn the quarter-wise base predictor $h^t : \Theta \times \mathcal{S}^t \to \mathcal{Y}$ for each quarter t using a training dataset. As pointed out, the academic state space $\mathcal{S}^t$ is extremely large since the completed courses and associated grades may differ significantly across students. Therefore, a major challenge in learning the base predictors is that of extracting valuable features $x(s_i^t)$ from the academic state that are amenable to efficient data processing and learning. Once the features are extracted, state-of-the-art machine learning techniques can be applied to learn the base predictors using a training set.

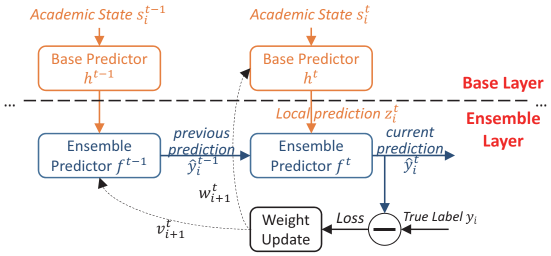

Figure 2: Ensemble Learning for Quarter t. Ensemble predictor f^t has two leafs f^{t-1} and h^t and makes a synthesized prediction $\hat{y}_i^t$ based on the leaf prediction $\hat{y}_i^{t-1}$ and z_i^t. Depending on the correctness of the prediction results, the weights of the leaf predictors are updated.

Feature Vector Construction

A student's academic state includes all courses that the student has completed and their grades and credits. A straightforward way to construct a feature x_i^t is to assemble all this information into a vector in a unified format for all students. However, since different students can take different courses, x_i^t must also include information for courses that student i have not taken, thereby making x_i^t a large but sparse vector. Suppose there are a total number of N courses (including curriculum courses and all possible extra-curriculum courses). For each course, there are two entries in the vector x_i^t, with one entry indicating the earned credit and the other indicating the earned grade. If student i has not taken a particular course, then the earned credit is 0 and the earned grade is *NULL*. Therefore, the feature vector size is $2N$.

An obvious drawback of this approach is that the feature vector can be very large. In a typical four year college program, there are often more than 50 curriculum courses, let alone numerous extra-curricular courses. Thus the feature vector size will be on the order of hundreds. Given the fact that education datasets are often relatively small since the number of students enrolled in the same program is limited, training a reasonably good base predictor can be very difficult. Moreover, using a feature vector at such a fine level of granularity may also reduce the prediction performance since much noise may be included.

We reduce the feature vector in two steps as follows.

Course Clustering using Domain Knowledge. The first step to reduce the feature vector size is to cluster the courses using educational domain knowledge. For instance, all math courses can be put in one cluster and all curriculum courses can be put in another. Notice that a course can belong to multiple clusters in our setting. For each cluster, we obtain

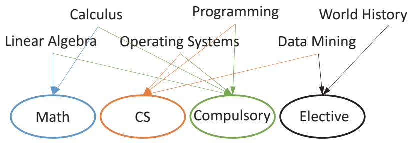

Figure 3: Illustration of Course Clustering using Domain Knowledge.

the average grade of completed courses in the cluster and the total credits that have been earned by the student. Suppose that M clusters are used, then the size of the feature vector is $2M$. Figure 3 shows an illustration of course clustering using educational domain knowledge.

Feature Selection. Given the constructed course clusters, feature selection is carried out in order to (1) further reduce the feature vector size and (2) gain insights on the most important features that affect the graduation time. A multitude of feature selection methods can be utilized to perform this task. Detailed feature selection results and discussions will be provided in the experiments section.

Learning Ensemble Predictors

We consider a stochastic setting where the student data arrive in sequence $i = 1, 2, ...$ Such a stochastic setting is suitable for both offline training and online updating. Given an offline training dataset, the student arrival sequence can be easily generated according to any random process. In the online scenario, the ensemble predictors are able to keep improving itself using the new student data. Each student belongs to a student group depending on the static feature of the student that does not change over quarters, such as the pre-college traits. The student groups can be created by a variety of state-of-the-art clustering algorithms. Let $g_i \in \mathcal{G}$ be the group of student i and $\mathcal{G}$ be the group space. For instance, $\mathcal{G}$ can consist of a high SAT score group and a low SAT score group.

Our ensemble predictors are learned based on the EWAF algorithm, taking into account the student context information and evolving progress. Specifically, each base predictor h^t is associated with a weight vector $\boldsymbol{w}_i^t$ and each ensemble predictor f^t is associated with a weight vector $\boldsymbol{v}_i^t$, both of which have size $|\mathcal{G}|$ and are changing over the student index i. All weight vectors are initialized to be $(1, ..., 1)$. For student i at the beginning of quarter t, the ensemble predictor f^t makes a prediction $\hat{y}_i^t$ based on a weighted average of the base prediction result z_i^t and the previous quarter ensemble prediction result $\hat{y}_i^{t-1}$. Depending on whether the prediction is a regression problem (e.g. to predict when to graduate so $\mathcal{Y}$ is a continuous space) or a classification problem (e.g. to predict whether or not to graduate on time so $\mathcal{Y}$ is a binary space), the weighted averaging is deterministic or probabilistic as follows:

Regression: The ensemble prediction is

$$\hat{y}_i^t = \frac{\boldsymbol{v}_i^{t-1}(g_i)\hat{y}_i^{t-1} + \boldsymbol{w}_i^t(g_i)z_i^t}{\boldsymbol{v}_i^{t-1}(g_i) + \boldsymbol{w}_i^t(g_i)} \tag{1}$$

Classification: The ensemble prediction is $\hat{y}_i^{t-1}$ (or $\hat{z}_i^t$) with probability

$$\frac{\boldsymbol{v}_i^{t-1}(g_i)}{\boldsymbol{v}_i^{t-1}(g_i) + \boldsymbol{w}_i^t(g_i)} \quad (or \ \frac{\boldsymbol{w}_i^t(g_i)}{\boldsymbol{v}_i^{t-1}(g_i) + \boldsymbol{w}_i^t(g_i)}) \tag{2}$$

Depending on the true label y_i (i.e. the true graduation time or whether the student graduated on time or not) of student i, the weights of the base predictor and the previous quarter ensemble predictor are updated according to their cumulative loss for group g_i students. Specifically, the cumulative loss for base predictor h^t up to student n for group g is

$$L_n(h^t|g) = \sum_{i=1}^{n} l(z_i^t, y_i)\mathbf{1}\{g_i = g\} \tag{3}$$

and the cumulative loss for the previous quarter ensemble predictor f^{t-1} up to student n for group g is

$$L_n(f^{t-1}|g) = \sum_{i=1}^{n} l(\hat{y}_i^{t-1}, y_i)\mathbf{1}\{g_i = g\} \tag{4}$$

where $l(y, y')$ is a loss function that characterizes the prediction loss. For instance, for the regression case $l(y, y') = (y - y')^2$ and for the classification case $l(y, y') = \mathbf{1}\{y \neq y'\}$.

The weights then are constructed using the cumulative loss based on an exponential function as follows

$$\begin{aligned} \boldsymbol{w}_{i+1}^t(g) &= \exp(-\eta_i L_i(h^t|g)) \\ \boldsymbol{v}_{i+1}^{t-1}(g) &= \exp(-\eta_i L_i(f^{t-1}|g)) \end{aligned} \tag{5}$$

where η_i is a sequence of input constants. Intuitively, the predictor with a larger cumulative loss will be assigned with a smaller weight. The complete proposed Ensemble-based Progressive Prediction (EPP) algorithm is given in Algorithm 1.

Algorithm 1 Ensemble-based Progressive Prediction (EPP)

```
**Initialization**: $L(h^t) = 0, L(f^t) = 0, \forall t.$
for each student $i$ do
    Observe background $\theta_i$, student group $g_i$
    for quarter $t = 1$ to $T$ do                ▷ Prediction Phase
        Observe academic state $s_i^t$
        Receive prediction $\hat{y}_i^{t-1}$ from $f^{t-1}$
        Base predictor $h^t$ predicts $z_i^t = h^t(\theta_i, s_i^t)$
        Ensemble predictor $f^t$ predicts
            $\hat{y}_i^t = f^t(\hat{y}_i^{t-1}, z_i^t | \boldsymbol{v}_i^{t-1}, \boldsymbol{w}_i^t)$
    end for
    Observe true label $y_i$.
    for quarter $t = 1$ to $T$ do                ▷ Update Phase
        Compute prediction loss $l(\hat{y}_i^t, y_i)$ and $l(z_i^t, y_i)$
        Update $L_i(h^t|g_i) \leftarrow L_{i-1}(h^t|g_i) + l(z_i^t, y_i)$
          $L_i(f^{t-1}|g_i) \leftarrow L_{i-1}(f^{t-1}|g_i) + l(\hat{y}_i^t, y_i)$
        Update weights $\boldsymbol{w}_{i+1}^t$ and $\boldsymbol{v}_{i+1}^t$ according to (5)
    end for
end for
```

Performance Analysis

In this section, we characterize the performance of the proposed progressive predictor. We study only the case $|\mathcal{G}| = 1$ since the extension is straightforward. We will focus on the ensemble predictor for a particular quarter t and compare it with the best base predictor among quarters $1, 2, ..., t$ in hindsight. Once the performance of ensemble predictor f^t is characterized, characterizing the overall performance of the progressive predictor is straightforward. The performance metric is regret. However, since the ensemble prediction is deterministic in the regression case while probabilistic in the classification case, we define and analyze the regret separately for these two cases.

Regression Case

Let $h^*(t) = \arg\min_{h^\tau:\tau\in\{1,...,t\}} L_n(h^\tau)$ be the best base predictor in hindsight restricted to the first t quarters and $L_n^{*,t} = L_n(h^*(t))$ be the corresponding minimal cumulative loss. The regret of ensemble predictor f^t up to student n is defined as $\text{Reg}^t(n) = L_n(f^t) - L_n^{*,t}$.

Proposition 1. *Assume that $l(\cdot,\cdot)$ is convex in its first argument. When the EPP algorithm runs with parameter $\eta_i = \sqrt{8(\ln 2)/i}$, then for any number n of students, the regret of ensemble predictor f^t satisfies Reg$^t(n) \leq t\left(2\sqrt{n\ln 2/2} + \sqrt{\ln 2/8}\right)$.*

Proof. The proof is largely based on Theorem 2.3 in (Cesa-Bianchi and Lugosi 2006), which can be used to establish bounds on the performance difference between f^t and the best among h^t and f^{t-1} when we restrict ourselves to the ensemble learning problem at quarter t. In particular, we have

$$\begin{aligned} &L_n(f^t) - \min\{L_n(f^{t-1}), L_n(h^t)\} \\ \leq& 2\sqrt{n\ln 2/2} + \sqrt{\ln 2/8} \end{aligned} \tag{6}$$

By considering the bounds for quarter $t-1$, we have

$$\begin{aligned} &L_n(f^t) - L_n(h^{t-1}) \\ =&L_n(f^t) - L_n(f^{t-1}) + L_n(f^{t-1}) - L_n(h^{t-1}) \\ \leq& 2(2\sqrt{n\ln 2/2} + \sqrt{\ln 2/8}) \end{aligned} \tag{7}$$

By induction, we obtain the claimed result. □

Classification Case

Since the prediction output in the classification case is randomly sampled according to a distribution, we define regret is a slightly different way. Instead of using the realized cumulative loss, we use the expected cumulative loss when defining the regret: $\text{Reg}^t(n) = \text{E}[L_n(f^t)] - L_n^{*,t}$. Because of the probabilistic prediction, the space of predictions and the loss functions are not convex in the classification case and hence, results of Theorem 2.3 in (Cesa-Bianchi and Lugosi 2006) are no longer applicable. Theorem 2 in (Tekin, Yoon, and van der Schaar 2015) establishes a similar result for the classification case, which is utilized in our regret analysis.

Proposition 2. *When the EPP algorithm runs with parameter $\eta_i = \sqrt{\ln 2/i}$, the regret of ensemble predictor f^t satisfies Reg$^t(n) \leq t\left(2\sqrt{n\ln 2}\right)$.*

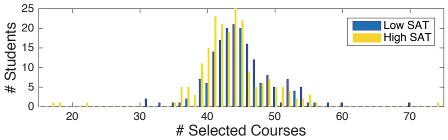

Figure 4: Distribution of Number of Selected Courses

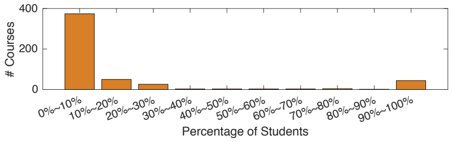

Figure 5: Courses Selected by Student Percentage. For example, there are 374 courses, each of which is selected by fewer than 10% students.

Proof. Consider the ensemble learning problem at quarter t, Theorem 2 in (Tekin, Yoon, and van der Schaar 2015) can be used to show

$$\text{E}[L_n(f^t)] - \min\{L_n(f^{t-1}), L_n(h^t)\} \leq 2\sqrt{n\ln 2} \tag{8}$$

Then the proposition can be similarly proved as in Proposition 1. □

Now we have shown that the regret bounds for both the regression and the classification cases are sublinear in the number of students n. This implies that the average regret tends to zero if $n \to \infty$, i.e. $\lim_{n\to\infty} \frac{1}{n}\text{Reg}^t(n) \to 0$. These bounds provide a worst-case performance guarantee for the ensemble predictor, stating that the ensemble predictor is no worse than the best base predictor in hindsight asymptotically. However, we point out that the ensemble predictor often performs much better than all base predictors in practice as we will show in the experiments.

Moreover, these propositions in fact highlight the importance of taking the student's evolving progress into account for the student future performance prediction. Consider a particular quarter t. The base predictor h^t uses only the current academic state but not the previous academic states to make predictions whereas the ensemble predictor f^t implicitly uses all past academic states. The cumulative loss $L_n(h^t)$ of h^t clearly is no less than $L_n^{*,t}$ for any t. Since $L_n(f^t)$ is asymptotically no more than $L_n^{*,t}$, it can be concluded that using the student's evolving past progress improves the prediction performance by using our algorithm.

Experiments

Dataset

Student data used to test our algorithm is collected from UCLA Mechanical and Aerospace Engineering Department.

Table 1: Dataset Statistics

	On Time	Not On Time
Average GPA	3.33	3.17
Average Credits	172.4	157.9

The dataset has 367 anonymized students enrolled in the same program. The data of each student contains the student's pre-college traits (high school GPA and SAT scores), the courses (lectures, labs) that the students take in each academic quarter, the course credits and the obtained grades. We consider only the fall, winter and spring quarters but not the summer quarter since very few students take classes in summer quarters. Thus, we say that a student graduate on time if he/she graduates within 12 quarters and does not take any more courses beyond the first 12 quarters. In this dataset, the on-time graduation rate is 58%. Table 1 shows that students who graduated on time also had a higher GPA and earned more credits on average. However, on one hand, using only the average GPA and total credits ignores the diversity of courses taken by students and it can be much easier to earn a high grade in some courses than others. On the other hand, there are numerous possibilities of the courses selected by students. Figure 4 illustrates the distribution of number of courses selected by students and Figure 5 shows the number of courses that are selected by a certain percentage of students. It can been seen that although the students come from the same program, the courses that students take are extremely diverse. The total number of distinct courses among all students is 453, which is much more than the average number 44 of courses selected by students.

Due to space limitation, we show results for the classification task only in which we predict whether a student can graduate on time or not.

Important Features

We constructed a feature vector of size 22, which includes the high school GPA, the SAT scores and the average grade and total credits for 10 course clusters (math, chemistry, physics, elementary major, advanced major, related major, senior design, elective, compulsory, overall). To find out the most significant features, we used the standard LASSO-based feature selection method (Tang, Alelyani, and Liu 2014). Specifically, we solve the following loss minimization problem where A is the feature weight vector: $\min_A \|A^T\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \lambda\|A\|_1$. $\|A\|_1$ is included in the objective to induce sparsity in the feature weight vector. Features that have non-zero value in A are considered significant. The significant features vary across quarters, which are illustrated in Figure 6. It turns out that (1) SAT scores are more important static features than the high school grades; (2) Course grades are more important than credits; (3) Elementary and advanced major courses become important in the later stage of the program when students begin to take them; (4) senior design is the most important in the final quarter.

Performance

We compare the performance of our EPP algorithm with the following benchmarks

- *Using only current state (Base)*: This is simply using only the base predictor h^t to predict in quarter t without utilizing the evolving student performance. We implemented SVM (with various kernels) and KNN for the base predictors. KNN yields the highest accuracy and hence we report results only for KNN.
- *Using only evolving GPA (GPA)*: A binary classifier is trained using only the overall GPA up to quarter t to predict in quarter t.
- *Using only evolving total credit (Credit)*: A binary classifier is trained using only the total credits earned up to quarter t to predict in quarter t.

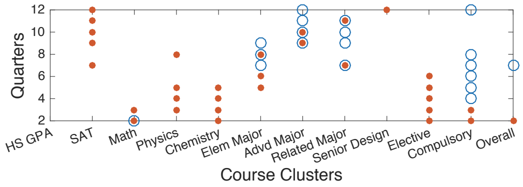

Figure 6: Significant Features. (Circle - credits; Dot - grade)

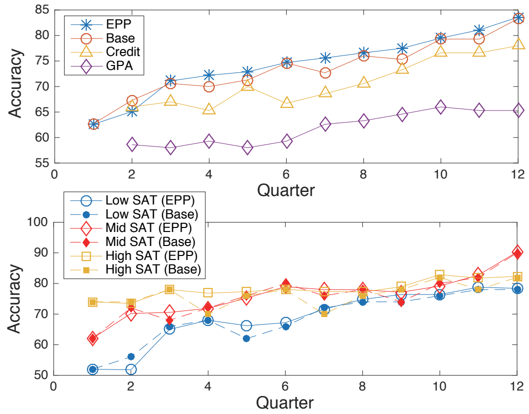

Figure 7: Prediction Accuracy over Quarters

In the simulations, we created three student groups (high/mid/low SAT scores). Half of the students were used as the training data and the remaining students were used as the testing data. Figure 7 shows the prediction accuracy over quarters and breaks down for different student groups. As we can see, all algorithms generally improve their accuracy over the quarters since more information is accumulated. Moreover, the proposed EPP outperforms all the benchmark algorithms.

Conclusion

In this paper, we proposed a novel algorithm for predicting student's performance in college programs given his/her current academic records. Our data-driven approach can be used together with other pedagogical methods for evaluating student's performance and provide valuable information for academic advisors to recommend subsequent courses to students and carry out pedagogical interventions measures if necessary.

References

Bekele, R., and Menzel, W. 2005. A bayesian approach to predict performance of a student (bapps): A case with ethiopian students. *algorithms* 22(23):24.

Brinton, C. G., and Chiang, M. 2015. Mooc performance prediction via clickstream data and social learning networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, 2299–2307. IEEE.

Brunskill, E., and Russell, S. 2010. Partially observable sequential decision making for problem selection in an intelligent tutoring system. In *Educational Data Mining 2011*.

Cen, H.; Koedinger, K.; and Junker, B. 2006. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International Conference on Intelligent Tutoring Systems*, 164–175. Springer.

Cesa-Bianchi, N., and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge university press.

Complete College America. 2014. Four-year myth: Making college more affordable. http://completecollege.org/wp-content/uploads/2014/11/4-Year-Myth.pdf.

Cucuringu, M.; Marshak, C.; Montag, D.; and Rombach, P. 2016. Rank aggregation for course sequence discovery. *arXiv preprint arXiv:1603.02695*.

Feng, M.; Heffernan, N.; and Koedinger, K. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction* 19(3):243–266.

Hoiles, W., and van der Schaar, M. 2016. Bounded off-policy evaluation with missing data for course recommendation and curriculum design. In *Proceedings of The 33rd International Conference on Machine Learning*, 1596–1604.

KDD Cup. 2010. Educational data minding challenge. https://pslcdatashop.web.cmu.edu/KDDCup/.

Lee, J. I., and Brunskill, E. 2012. The impact on individualizing student models on necessary practice opportunities. *International Educational Data Mining Society*.

Mandel, T.; Liu, Y.-E.; Levine, S.; Brunskill, E.; and Popovic, Z. 2014. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 1077–1084.

Marquez-Vera, C.; Romero, C.; and Ventura, S. 2010. Predicting school failure using data mining. In *Educational Data Mining 2011*.

Meier, Y.; Xu, J.; Atan, O.; and van der Schaar, M. 2015. Personalized grade prediction: A data mining approach. In *Data Mining (ICDM), 2015 IEEE International Conference on*, 907–912. IEEE.

Meier, Y.; Xu, J.; Atan, O.; and van der Schaar, M. 2016. Predicting grades. *IEEE Transactions on Signal Processing* 64(4):959–972.

Pardos, Z. A., and Heffernan, N. T. 2010. Using hmms and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. *Journal of Machine Learning Research W & CP*.

Tang, J.; Alelyani, S.; and Liu, H. 2014. Feature selection for classification: A review. *Data Classification: Algorithms and Applications* 37.

Tekin, C.; Yoon, J.; and van der Schaar, M. 2015. Adaptive ensemble learning with confidence bounds. *arXiv preprint arXiv:1512.07446*.

Thai-Nghe, N.; Drumond, L.; Horváth, T.; Schmidt-Thieme, L.; et al. 2011. Multi-relational factorization models for predicting student performance. In *Proc. of the KDD Workshop on Knowledge Discovery in Educational Data*. Citeseer.

Thai-Nghe, N.; Horváth, T.; and Schmidt-Thieme, L. 2010. Factorization models for forecasting student performance. In *Educational Data Mining 2011*.

The White House. 2016. Making college affordable. https://www.whitehouse.gov/issues/education/higher-education/making-college-affordable.

Toscher, A., and Jahrer, M. 2010. Collaborative filtering applied to educational data mining. *KDD cup*.

Wang, Y.-h., and Liao, H.-C. 2011. Data mining for adaptive learning in a tesl-based e-learning system. *Expert Systems with Applications* 38(6):6480–6485.

Xu, J.; Xing, T.; and van der Schaar, M. 2016. Personalized course sequence recommendations. *IEEE Transactions on Signal Processing* 64(20):5340–5352.

Yu, H.-F.; Lo, H.-Y.; Hsieh, H.-P.; Lou, J.-K.; McKenzie, T. G.; Chou, J.-W.; Chung, P.-H.; Ho, C.-H.; Chang, C.-F.; Wei, Y.-H.; et al. 2010. Feature engineering and classifier ensemble for kdd cup 2010. In *Proceedings of the KDD Cup 2010 Workshop*, 1–16.