Next-Term Student Performance Prediction: A Recommender Systems Approach

Mack Sweeney
George Mason University
msweene2@gmu.edu

Huzefa Rangwala
George Mason University
rangwala@cs.gmu.edu

Jaime Lester
George Mason University
jlester2@gmu.edu

Aditya Johri
George Mason University
ajohri3@gmu.edu

An enduring issue in higher education is student retention to successful graduation. National statistics indicate that most higher education institutions have four-year degree completion rates around 50%, or just half of their student populations. While there are prediction models which illuminate what factors assist with college student success, interventions that support course selections on a semester-to-semester basis have yet to be deeply understood. To further this goal, we develop a system to predict students' grades in the courses they will enroll in during the next enrollment term by learning patterns from historical transcript data coupled with additional information about students, courses and the instructors teaching them.

We explore a variety of classic and state-of-the-art techniques which have proven effective for recommendation tasks in the e-commerce domain. In our experiments, Factorization Machines (FM), Random Forests (RF), and the Personalized Linear Multiple Regression model achieve the lowest prediction error. Application of a novel feature selection technique is key to the predictive success and interpretability of the FM. By comparing feature importance across populations and across models, we uncover strong connections between instructor characteristics and student performance. We also discover key differences between transfer and non-transfer students. Ultimately we find that a hybrid FM-RF method can be used to accurately predict grades for both new and returning students taking both new and existing courses. Application of these techniques holds promise for student degree planning, instructor interventions, and personalized advising, all of which could improve retention and academic performance.

Keywords. matrix factorization; grade prediction; cold-start; recommender system; educational data mining; regression

1. Introduction

An enduring issue in higher education is student retention to successful graduation (Aud and Wilkinson-Flicker, 2013). The 2001 National Research Council report (Council, 2001) identified the *critical need* to develop innovative approaches to help higher-education institutions retain students, facilitate their timely graduation, and ensure they are well-trained and workforce ready in their field of study. Despite all efforts made since that time, the six-year graduation rate has increased by only 2% from the roughly 57% number observed in 2001 (Grace Kena et al., 2015). Meanwhile the four-year graduation rate remains just below 40%. The question then, is how can we increase student retention and knowledge acquisition? A highly referenced report by Vincent Tinto (2006) suggests that research on retention has made great advances, but effective implementation efforts by universities are few and far between, and more recent investigations conclude the same (Tinto, 2012). Some notable exceptions were detailed by The Educational Trust (Kevin Carey, 2005), in a report that identified three key characteristics of successful retention-increasing programs. The first two are student engagement and a genuine emphasis on teaching and learning. The third, and the one we focus on most in this study, is *effective utilization of new data systems to monitor student progress and identify key aspects of student success*.

The ability to predict student grades in future enrollment terms provides valuable information to aid students, advisors, and educators in achieving the mutually beneficial goal of increased student retention. This information can be used to help students choose the most suitable majors, properly blend courses of varying difficulty in a semester's schedule, and indicate to advisors and educators when students need additional assistance. Early identification of at-risk students is a key aspect of preventing them from becoming discouraged and dropping out (Grayson et al., 1998). Incorporation of next-term grade predictions in early-warning systems can increase the number of such students correctly identified, potentially increasing retention rates significantly. Furthermore, successful prediction models will yield valuable insights into what factors impact student success across various subpopulations. These insights can inform policy decisions to increase student engagement to further increase retention.

In this paper, we develop a system that leverages state-of-the-art recommender systems techniques to predict students' course grades for the next enrollment term in a traditional university setting [1]. Students take courses over a sequence of academic terms. During each term, students enroll in one or more courses and earn letter grades in the range A-F for each. Our dataset consists of grades from previous terms, which we call historical transcript data, coupled with student demographics, course characteristics, and instructor information. Given this data, our task is to predict the grades students will obtain in the courses they will take in the next term.

With this problem formulation, the next-term student grade prediction problem becomes quite similar to recommendation problems such as rating prediction and next-basket recommendation. Throughout the years, a variety of methods have been used for recommendation in the e-commerce domain (Bell et al., 2007). These recommender methods typically fall into one of two categories, depending on the attributes exploited for prediction: collaborative filtering (CF) and content-based (CB) methods. CF models use only the user-item rating matrix for predictions. In our setting, this is the user-course grade matrix with grades for each user in all courses he or she has completed. CB methods also have access to rating data but model it as an additional feature and incorporate features from either user profiles, item descriptions, or both. For grade prediction, these features consist of student demographics, performance history, and course features. These are collectively referred to as *content features*.

The purpose of the study presented in this paper was to apply state-of-the-art recommender systems techniques to the task of student performance prediction. We hoped to identify strengths and weaknesses, as well as synergies with other techniques commonly employed in the higher

[1] Source available at: `https://github.com/macks22/ntsgp`

education literature. We explore a variety of recommender system methods that leverage content features, blending CF and CB modeling ideas in powerful hybrid models. We also add data from students who transfer to the university and use various feature importance metrics to explore the different student characteristics with regards to course performance. This final analysis yields several interesting insights regarding the importance of instructors and the differing factors that affect transfer and non-transfer (native) student performance.

The task of predicting students' grades in the coming term is complicated by the ever-expanding volume of data from increasing student enrollments and by the continually shifting characteristics of the overall student population. Furthermore, recommender systems typically have issues predicting for previously unseen users and items; these issues are collectively called the *cold-start problem*. Our experiments show that Factorization Machines (FMs), Random Forests (RFs), and the Personalized Linear Multiple Regression (PLMR) model achieve the lowest prediction error. While these methods also struggle with the cold-start problem, we found that incorporation of key content features results in improved predictions from at least one of the top three methods for each cold-start scenario. Application of a novel feature selection metric for the FM model greatly improves its accuracy. With the improved FM model, we were also able to devise a hybrid FM-RF method that outperforms all three individual methods, exploiting the strengths of both models to overcome the cold-start problem.

2. Literature Review

Recommendation methods include nearest neighbor approaches (Linden et al., 2003) (Adamic et al., 2001) (Huete et al., 2012), matrix factorization (MF) for collaborative filtering (Koren and Bell, 2011) (Shan and Banerjee, 2010), restricted Boltzmann machines (Salakhutdinov et al., 2007), and topic modeling methods (Alsumait et al., 2010), among others. Much of the research pertaining to recommendation tasks has been conducted in the domain of e-commerce. In particular, the task of movie recommendation was widely studied during the Netflix Prize competition (Bell et al., 2007). Many companies have come to rely heavily on large-scale recommender systems for reducing information overload and targeting advertisements in industrial settings (Linden et al., 2003) (Kanagal et al., 2012) (Shani et al., 2005). While a variety of more traditional machine learning and case-based methods have been employed for student performance prediction (Romero and Ventura, 2010), the application of state-of-the-art recommender systems techniques to learner performance prediction is a relatively recent development (Drachsler et al., 2015).

In this paper we study the application of recommender system technology to student grade prediction within a traditional university setting. Similar performance prediction techniques have been explored, but mostly in the context of online learning environments, such as Massive Open Online Courses (MOOCs) (Romero et al., 2008) (Romero et al., 2013) (Peña-Ayala, 2014) (Liyanagunawardena et al., 2013) (Evans et al., 2016) and Intelligent Tutoring Systems (ITS) (Thai-Nghe et al., 2011) (Thai-Nghe et al., 2011) (Nedungadi and Smruthy, 2016) (Thai-Nghe et al., 2012). The goal in the present study is to predict grades in a traditional university learning environment. Online learning environments provide a variety of detailed student behavior data that is not available in traditional learning environments. The timeline of those studies is also more granular. In MOOCs, studies on predicting student performance (PSP) usually seek to predict grades for homeworks, quizzes, and exams within a single course. Some studies do span multiple courses, but the timelines are often a single year or less (Peña-Ayala, 2014).

Despite the differences, there are some interesting similarities between the present work and recent work in predicting correct first attempt (CFA) in ITS. CFA is a binary indicator: 0 if a student gets the problem/task correct on the first try and 1 otherwise. These binary targets are aggregated to produce real-valued ”ratings” in the $[0, 1]$ range. Thai-Nghe (2011) utilizes standard MF and slightly customized tensor factorization (TF) models. Results demonstrate that MF and TF methods achieve good predictive performance when compared with logistic regression and simple baselines. The MF models used in that study are subsumed by the FM model used in the present study. The TF model was employed to capture 3-way user-item-time interactions. While we do explore the utility of 1- and 2-way interaction terms, we do not explore 3-way interactions in this study. This would be an interesting direction for future work.

Both Thai-Nghe et al. (2011) and Nedungadi and Smruthy (2016) apply multi-relational matrix factorization (MRMF) models to CFA prediction. This method simultaneously performs MF on several ”rating” matrices for complementary relations, such as ”student solves a problem”, ”problem requires a skill”, and ”student has a skill.” Jointly decomposing several matrices into partially shared factorized matrices is empirically shown to more effectively regularize the entities in the primary relation, yielding improved predictions. Applying unequal weights (WMRMF) to these joint learning objectives can further improve results. Adding bias terms to both the MRMF and WMRMF models can achieve even further improvements. Thia-Nghe et al. (2012) also apply the FM model to the task of CFA prediction. However, they limit their data to the user-item matrix and learn the model with stochastic gradient descent (SGD). In this setting, the FM model reduces to a regularized Singular Value Decomposition (SVD) with bias terms and a global intercept. We studied the ability of this and other similar models for predicting grades on a subset of the present dataset in a prior work (Sweeney et al., 2015).

ITS researchers have also been very interested in student modelling and knowledge tracing. In a seminal paper, Corbett and Anderson (1994) presented an analysis whose task was modeling the changing knowledge states of students during skill acquisition. These student models often form core components of ITS systems. Desmarais and Baker (2011) provide a broad review of ITS systems in. One example of recent state-of-the-art includes Xu and Mostow’s (2012) logistic regression dynamic Bayes net (LR-DBN) model for tracing multiple student subskills over time. They also recently conducted a morphological analysis of prominent student models, surveying recent methods and identifying gaps to be filled (Xu and Mostow, 2015). Barnes and Stamper (2008) demonstrated another interesting direction with their use of Markov Decision Processes (MDPs) for automatically generating hints for an intelligent logic proof tutor. Baker et al. (2008) also developed systems that can adapt and intervene based on predictions of future student performance, and they were able to show these interventions were effective in improving students’ experiences. It would be very interesting to explore the modeling techniques leveraged in these studies for the task of grade prediction in future work. The present study makes no attempt to model learner knowledge states or to provide interactive feedback during activities.

There are some recent studies which also apply cutting-edge regression techniques to grade prediction in university settings, which we describe in the next few paragraphs. In one such study, comments from learning reflection assignments in a course are used as features for grade prediction (Sorour et al., 2015). Latent Semantic Analysis (LSA) is used to extract topics and reduce dimensionality of the comments. Then k-means is used to cluster the comments into per-grade clusters. Rhodes et al. (2013) also leverage lexical features from student self-reflection for performance prediction. By analyzing these features together with student effort features (time spent writing solutions), they find that self-explanation vocabulary correlates with amount of effort expended. The features in these studies consist only of bag-of-words term vectors. The present study does not leverage any such text features, since none were readily available in our dataset.

In our previous work, we studied the next-term grade prediction task without incorporating content features, i.e. exploring only CF methods (Sweeney et al., 2015). The present effort expands upon this work to explore a variety of methods that leverage content features, blending CF and CB modeling ideas in powerful hybrid models. In another recent study by Elbadrawy et al. (2015), the authors devise a custom mixed-membership linear regression model (PLMR – mentioned above) to predict student grades in a traditional university setting. They use a variety of data, including grades and learning management system (LMS) features. Use of LMS data allows prediction of grades at the ”activity” level – individual homeworks, quizzes, or exams within a given course. The present study does not use data from or predict grades for individual activities. Instead, grades are predicted only at the course level using only course-level (rather than activity-level) information. We do, however, compare against the PLMR model in our study and find it to be one of the top three performers. In the present study, we compare more models on a larger dataset and conduct a more detailed analysis of feature importance, performance errors, and implications for educational applications. We also provide a thorough discussion of challenges faced in predicting for new courses and students and identify key differences between transfer and native student performance prediction.

3. Problem Formulation

Given a database of (student, course) dyads (i.e., pairs) with associated content features for the course, student, and course instructor, our goal is to predict grades for each student for the next enrollment term. More formally, we have n students and m courses, comprising an $n \times m$ sparse grade matrix G, where $\{G_{ij} \in \mathbb{R} \mid 0 \leq G_{ij} \leq 4\}$ is the grade student i earned in course j. This is the primary source of information leveraged by matrix factorization techniques, such as Singular Value Decomposition (SVD). For those methods, the task is often cast as a matrix completion problem. However, we do not wish to complete the entire matrix. The work laid out in this paper assumes the student(s) have already selected a set or several possible sets of courses to take in the coming term. Our goal is to provide feedback to students, advisors, and educators regarding the selection(s) made or proposed. Recent research investigates how to predict the courses students will select in the coming term (Ognjanovic et al., 2016) and how to identify selections that satisfy degree requirements (Parameswaran et al., 2011). These efforts could be combined with ours to recommend additional courses or detect when students have made selections which will put them behind in their degree plan.

We consider each cell to be a (student, course) dyad and represent it as a feature vector $\boldsymbol{X}_{ij} \in \mathbb{R}^{1 \times p}$. The first n variables are a 1-of-n vector representing the one-hot-encoded student IDs and the next m are a 1-of-m vector for the course IDs. The rest of the p total features consist of content features generated by student i taking course j. We train our models on all feature vectors $\boldsymbol{X}_{ij}$ preceding the current term and predict grades $\hat{G}_{ij}$ for all feature vectors $\boldsymbol{X}_{ij}$ in the current term. This setup is critical for avoiding data leakage – a severe experimental misstep that can inflate real-world significance of performance metrics (Pardos et al., 2012). For fair and effective evaluation of the proposed methods, we train one model per academic term in the dataset, and we use that model to predict only for this ”current term.” All results presented in this paper represent an aggregate of predictions obtained with this sequential train-predict loop.

4. Dataset Description

The data used for this study comes from a public university [2] with an enrollment of 33,000 students as of Fall 2014 (gmu, 2014). Observations begin in Summer 2009 and continue until Spring 2014. Each year there is a Spring, Summer, and Fall term; these terms span from January to May, May to August, and August to December. Hence our dataset contains a total of 15 terms. All students whose cohorts pre-date Summer 2009 were excluded from the data. After preprocessing, there are 30,754 students declared in one of 144 majors, each of which belongs to one of 13 colleges. 584,179 (65.29%) of these are transfer students and the other 310,557 are non-transfer (34.71%). During this time period, these students have taken 9,085 unique courses, each of which is classified as one of 161 disciplines and taught by one of 6,347 instructors. After discarding records with no grades or grades which do not translate to the A-F scale (such as withdrawals and audits), we have 894,736 dyads. 584,179 (65.29%) of these are transfer students and 310,557 (34.71%) are non-transfer (native) students. All this data was collected and anonymized in accordance with Institutional Review Board (IRB) policies.

The A-F letter grades are discrete, but they have ordinal equivalents in the range 0-4. This mapping is common in American universities, but for those unfamiliar, it proceeds as follows. An F is mapped to 0, a D to 1, a C- to a 1.67. From there, we have C, C+, B-, B, B+, A-, and A, which correspond to the numbers from 2 to 4 in increments of $\frac{1}{3}$. As a result of the ordinal mapping, this problem could be cast as either classification or regression. We explored both options in our preliminary experiments. In particular, we looked at using kNN, a Gaussian Mixture Model (GMM), Support Vector Machine (SVM), boosted decision trees, random forests, standard decision trees, and the FM. All of these classification methods performed poorly compared to the regression techniques; we believe this is because classification models fail to capture the ordinal nature of the data. In order to properly capture this ordering information, we cast the problem as a regression task.

4.1. Content Features

For each dyad, we have a variety of student, course, and instructor features, either categorical or real-valued. For each of these three entities, we have unique identifiers and a variety of other information. For students, we have demographics data, such as age, race, sex, high school CEEB code and GPA, zip code, and 1600-scale SAT scores. For each dyad, we have the declared major of the student and the grade earned. For each term, we have the GPA from the previous term as well as the cumulative GPA. We have the number of credit hours the student is enrolled in during the current term and the number of credit hours attempted up to the current term. We also have an academic level obtained from credit hours attempted. Finally, we annotate each term for each student with that student's relative term number. This feature reflects the number of terms the student has taken courses for.

There is also a variety of data for courses. Each course belongs to a particular discipline, is worth a fixed number of credit hours, and is assigned a particular course level. For each term, we have the aggregate GPA of the course from the previous term as well as the cumulative aggregate GPA over all terms the course has previously been offerred (in our dataset). We have the number of students enrolled in all sections during this term, as well as the total number of students enrolled for all prior terms the course has been offered. In addition to information

[2] George Mason University (GMU)

Table 1: Cold Start Proportion By Academic Term

Term	Term #	Dyads	NCS	CS	% CS
'09 Summer	0	252	0	252	100.00
'09 Fall	1	62,074	813	61,261	98.69
'10 Spring	2	41,075	17,545	23,530	57.29
'10 Summer	3	3,577	3,165	412	11.52
'10 Fall	4	80,287	21,685	58,602	72.99
'11 Spring	5	64,295	37,358	26,937	41.90
'11 Summer	6	7,043	6,734	309	4.39
'11 Fall	7	98,673	36,166	62,507	63.35
'12 Spring	8	80,200	53,587	26,613	33.18
'12 Summer	9	8,880	8,546	334	3.76
'12 Fall	10	107,511	51,130	56,381	52.44
'13 Spring	11	100,342	64,720	35,622	35.50
'13 Summer	12	10,678	10,388	290	2.72
'13 Fall	13	125,312	58,901	66,411	53.00
'14 Spring	14	104,537	76,640	27,897	26.69

CS: Cold-Start, NCS: Non Cold-Start

Term # denotes chronological ordering of academic terms in dataset. Cold-start dyads have either a new student, a new course, or both.

specific to the course itself, the course side information also includes instructor information. For each instructor, we have his or her classification (Adjunct, Full time, Part time, Graduate Research Assistant, Graduate Teaching Assistant), rank (Instructor, Assistant Professor, Associate Professor, Eminent Scholar, University Professor), and tenure status (Term, Tenure-track, Tenured). Transfer course records are mapped to GMU equivalents. Instructor information for these records is mostly absent. So we use the ID of the institution of origin to substitute for an instructor ID and leave out the other instructor features. A more detailed listing of the features used in this study is given in Appendix I: Feature Descriptions.

4.2. Cold Start Predictions

In the context of the next-term prediction task, cold-start records are defined as (student, course) dyads for which either or both the student and course appear in a term (prediction phase) but do not appear in any of the previous terms (training phase). In our dataset, 447,378 (50.00%) dyads are non-cold-start and 447,358 (50.00%) are cold-start. 41,843 (9.35%) of these are dyads for which both the student and the course are cold-start. 389,449 (87.06%) are student-only cold-start, and 16,066 (3.59%) are course-only cold-start. Table 1 breaks down the proportion of cold-start records by academic term. For instance, 98.69% of the dyads are cold-start for the Fall 2009 cohort. The only previous enrollment was in Summer 2009, so we know the other 1.31% of the dyads represent students who enrolled in the previous summer.

5. Methods

We explore three classes of methods for the next-term student grade prediction task. These are (1) simple baselines, (2) MF-based methods, and (3) common regression models. The first two classes are methods which do not incorporate content features. The last consists of traditional regression methods which must necessarily incorporate content features. Since the FM model can use any features, it falls into (3) when incorporating content features and (2) otherwise.

5.1. Simple Baselines

We devised three simple baselines to better understand the effect of leveraging three types of central tendencies in the data.

- *Uniform Random (UR)*: Randomly predict grades from a uniform distribution over the range [0, 4].
- *Global Mean (GM)*: Predict grades using the mean of all previously observed grades.
- *Mean of Means (MoM)*: Predict grades using an average of the global mean, the per-student mean, and the per-course mean.

The UR method illustrates the result of making predictions by randomly guessing. The GM method illustrates the informative value of the overall central tendency of the data, often called the *global intercept*. The MoM method takes this strategy one step further, also incorporating the per-student and per-course (row and column) averages. When compared to the GM method, this illustrates the added benefit of a small level of personalization and historical knowledge about the course difficulty. In the cold-start setting, we allow the MoM method to use whatever information is available. For a particular cold-start dyad, either the student or course may be present, but not both. If neither is present, it reduces to the GM method.

5.2. Matrix Factorization Methods

We look at three methods based on Matrix Factorization (MF):

- *Singular Value Decomposition (SVD)*.
- *SVD-kNN*: SVD post-processed with kNN (Arkadiusz Paterek, 2007).
- *Factorization Machine (FM)* (Rendle, 2012).

Each of these models attempt to capture the pairwise interaction of features by decomposing the feature space into a k-rank reduced subspace. This results in two sets of latent vectors: one for the courses ($v_j \in \mathbb{R}^k$), and one for the students ($v_i \in \mathbb{R}^k$), which can be thought of as latent course characteristics and latent student competencies or knowledge states for each of these general course characteristics, respectively. These latent feature vectors can be considered a less noisy, condensed representation of the student and course information. For SVD, each grade is simply predicted as the dot product of the latent student and course feature vectors. We call this the factorized 2-way interaction of the student with the course. This differs from a simple linear regression in its use of latent feature vectors to deal with sparsity in the data. It also differs in its use of 2-way interactions, which capture the effects that interactions of two predictor variables have on the target variable (the grade).

$$\hat{G_{ij}} = \sum_{f=1}^{k} v_{i,f} v_{j,f} = v_i^T v_j. \tag{1}$$

Paterek (Arkadiusz Paterek, 2007) showed that post-processing SVD with k-nearest neighbors (kNN) yields improved predictive performance. In essence, the predicted grade for a course is replaced with the predicted grade for the most similar course. Similarity is calculated as cosine similarity in the latent feature space. Though this method has not, to our knowledge, been analyzed formally, it has been shown to be effective in practice. It can be thought of as a cluster-based smoothing, where the clusters are computed from latent course characteristics.

Pure collaborative filtering (CF) methods such as SVD and SVD-kNN are unable to make predictions for cold-start records. Without any previous observations for a student or a course, no latent student competencies or latent course characteristics can be learned. Content-based (CB) methods handle course cold-start by incorporating features describing each item and/or student profiles such as demographics (Adomavicius and Tuzhilin, 2005). FMs can incorporate arbitrary content features while also leveraging the sparse student-course grade matrix. This capability means the FM is a hybrid model which can incorporate the data traditionally used by CB and CF methods in one model. Combined with suitable feature engineering, this allows FMs to subsume most state-of-the-art factorization-based recommender system models developed up to this point (Rendle, 2012).

We observe that the FM model is able to capture all of the information captured by the simple baselines as well as the information captured by SVD. In general, it captures the global central tendency, 1-way (linear) relationships between the predictors and the grade (bias terms), and 2-way factorized interactions between each predictor and the grade:

$$\hat{G_{ij}} = \overbrace{w_0}^{\text{global intercept}} + \overbrace{\sum_{l=1}^{p} w_l x_l}^{\text{1-way interactions}} + \overbrace{\sum_{l=1}^{p}\sum_{l'=l+1}^{p} x_l x_{l'} (v_l^T v_{l'})}^{\text{2-way factorized interactions}} \tag{2}$$

The FM model can be understood as an adaptation of second order polynomial regression (PR) for sparse data. This is particularly evident from comparing the FM formula with that of the second order PR model:

$$\hat{G_{ij}} = \overbrace{w_0}^{\text{global intercept}} + \overbrace{\sum_{l=1}^{p} w_l x_l}^{\text{1-way interactions}} + \overbrace{\sum_{l=1}^{p}\sum_{l'=l}^{p} x_l x_{l'} w_{l,l'}}^{\text{2-way interactions}} \tag{3}$$

We use overloaded notation here, representing both the 1-way interaction terms and the 2-way interaction terms with w and differentiating by the number of subscripts. There are two main differences between the FM and PR model. First, the 2-way interactions are not factorized – note the replacement of $v_l^T v_{l'}$ with $w_{l,l'}$. This means the model will be less able to deal with data sparsity and hence unlikely to learn patterns for categorical data effectively. Each interaction can only be learned if it occurs in the data. Consider the student and course interaction features. We can only learn a pattern for a student in a particular course if we observe the outcome of the student taking the course. Even if we were to keep outcomes from students retaking courses, this would not happen for the majority of courses. So by the time we have learned the interaction, it is unlikely to be useful to us; students already know what grade they got in courses they have completed. The second difference is the inclusion of squared terms, i.e. 2-way interactions between each term and itself. These terms were not included in the FM model for *properness* and *multilinearity*; for more formal details, see (Freudenthaler et al., 2009).

We compare SVD and SVD-kNN to the FM model trained using only the student-course grade matrix G. With this data, the model reduces to the sum of global (w_0), student (w_i), and course (w_j) bias terms and the factorized interaction of the student with the course ($v_i^T v_j$). This last term is the same dot product seen in the SVD model. The other terms capture information modeled by the naive baselines.

$$\hat{G_{ij}} = w_0 + w_i + w_j + v_i^T v_j. \tag{4}$$

We use the fastFM library (Bayer, 2015) for a fast implementation of the FM algorithm. The model parameters are learned with Gibbs sampling, which is a common Markov Chain Monte Carlo (MCMC) learning algorithm.

5.3. Common Regression Models

We tested four different regression models:

- *Random Forest (RF)*
- *Stochastic Gradient Descent (SGD) Regression*
- *k-Nearest Neighbors (kNN)*
- *Personalized Linear Multiple Regression (PLMR)*

We leveraged the scikit-learn library (Pedregosa et al., 2011) for the first three, which are common regression models. Parameter settings for each method were found using grid search on a sampled held-out set. These are listed in Appendix II: Model Parameter Settings. We briefly describe each of these models.

5.3.1. Random Forest

The Random Forest algorithm combines a group of random decision trees in a bagging ensemble (Breiman, 2001). A single non-random decision tree is constructed by discovering the most informative questions to ask in order to split all samples into groups with similar target attribute values. Each question splits the data into two or more groups by thresholding some feature of the samples. So we end up with a tree of decision nodes. For regression, the most informative questions are those that produce leaf nodes whose mean squared error (MSE) are minimal among all possible splits.

Tree construction stops once an additional split would not reduce MSE. Since this usually overfits, an early-termination criterion is often specified. This is usually maximum tree depth or minimum number of nodes at each leaf; we used the former for our experiments. Once built, the tree can be used for regression of new data samples. A sample is run through the decision-making sequence defined by the structure of the tree until reaching a leaf. Then the prediction is the mean of the grades of the samples at that node when training finished.

A random decision tree results from learning a decision tree on a bootstrap data sample and considering a random subset of features for each split. This "feature bagging" is done to reduce correlation between the trees. The Random Forest then combines many of these trees in a weighted averaging approach to make decisions regarding unseen data. This reduces the variance of the individual trees while retaining the low-bias.

5.3.2. Stochastic Gradient Descent (SGD) Regression

The SGD regression method learns a least squares linear regression fit under an L1 regularization penalty (absolute norm). The least squares fit minimizes the squared difference between actual and predicted grades, while the L1 regularization penalty encourages feature sparsity. In particular, unimportant parameters have a tendency to be pushed towards 0, so the L1 penalty operates as a kind of online feature selection. In contrast to ordinary least squares (OLS), this method is an approximate best fit, learned using stochastic gradient descent (SGD). SGD is a gradient-based optimization technique that updates the model parameters incrementally, rather than on the entire training set at once (which is what normal gradient descent would do). This reduces overfitting and significantly improves training time.

5.3.3. k-Nearest Neighbors

The k-Nearest Neighbors (kNN) algorithm is a classic method for clustering samples based on similarity. These clusterings can be used for regression in the following manner. First a pairwise distance metric is used to identify the k most similar neighbors among all dyads in the training set. The grade is then predicted using local interpolation of the targets associated with these neighbors. Many different distance metrics can be used; for our experiments, we use standard Euclidean distance:

$$Euclidean(\boldsymbol{X}_{i,j}, \boldsymbol{X}_{i',j'}) = \sqrt{\sum_{f=1}^{p} \left(\boldsymbol{X}_{i,j,f} - \boldsymbol{X}_{i',j',f}\right)^2}. \tag{5}$$

To predict a grade for a new dyad (i, j), the Euclidean distance from (i, j) to every dyad in the training set is computed. The k dyads (i', j') with the smallest distance are selected and placed into a set of neighbors $\mathcal{N}_{i,j}$. The grade for student i in course j is then predicted as the uniformly weighted average of these neighbors' grades.

5.3.4. Personalized Linear Multiple Regression

We also employ the more recently developed Personalized Linear Multiple Regression model from (Elbadrawy et al., 2015). The original model predicts a missing grade G_{ij} for student i in course j using:

$$\hat{G}_{ij} = s_i + c_j + \sum_{l=1}^{k} P_{il} \sum_{f=1}^{p} W_{lf} \boldsymbol{X}_{ijf} \tag{6}$$

$$= s_i + c_j + P_i W \boldsymbol{X}_{ij}, \tag{7}$$

where s_i is a bias term for student i, c_j is a bias term for course j. PLMR uses a linear combination of k regression models weighted on a per-student basis. P_i is the $1 \times k$ vector of model weights for student i and W is the $k \times p$ matrix of regression coefficients. $\boldsymbol{X}_{ij}$ is the feature vector generated by student i taking course j. The model is learned using the following objective function. Root Mean Squared Error (RMSE) is used as the loss function $\mathcal{L}(\cdot)$, P, and W are regularized using the squared Frobenius norm, and all parameters are constrained to be non-negative.

$$\underset{P,W,s,c}{\text{minimize}}\ \mathcal{L}(P,W,s,c) + \lambda_W(\|P\|_F^2 + \|W\|_F^2) \tag{8}$$

We implemented our own version of the PLMR model, adding a global intercept term w_0 and a regularization on the bias terms controlled by parameter λ_B. This global intercept captures sample-wide trends, just as in the GM method. The regularization term improves upon this method (particularly for cold-start scenarios) by avoiding overconfident bias term learning when observing small sample sizes. The resulting model and objective function are shown below. All parameters are contrained to be non-negative, as in the original model.

$$\hat{G}_{ij} = \overbrace{w_0}^{\text{new}} + s_i + c_j + P_i W \boldsymbol{X}_{ij} \tag{9}$$

$$\underset{P,W,s,c}{\text{minimize}}\ \mathcal{L}(P,W,s,c) + \lambda_W(\|P\|_F^2 + \|W\|_F^2) + \underbrace{\lambda_B(\|s\|_F^2 + \|c\|_F^2)}_{\text{new}} \tag{10}$$

6. Experimental Results and Discussion

6.1. Metrics

Evaluations are performed in terms of two common regression metrics: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE).

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(\hat{G}_{ij} - G_{ij})^2}{N}} \qquad MAE = \frac{\sum_{i=1}^{N}|\hat{G}_{ij} - G_{ij}|}{N}$$

RMSE penalizes severe prediction errors more heavily than small ones. Given our task, we would prefer not to declare a method the best if it performs very well for half the students and very poorly for the other half. Hence RMSE is the metric we use to compare methods. MAE allows us to understand the range of grades we might actually be predicting. If a student is actually going to get a B, and the MAE is 0.33, we expect our model to predict either a B-, B, or B+. All MAE measurements are accompanied by the standard deviation of the absolute errors. Note that both of these metrics measure error, so a smaller number is better.

6.2. Feature Preprocessing and Selection

For preprocessing, missing values for each real-valued feature were filled in using the medians. This was not done for the FM model, since it handles missing data without loss of performance and performs worse with the median-value imputation. After this step, the real-valued attributes were scaled using Z-score scaling. Finally, the predictions of all methods are post-processed to bound the grade predictions to the range [0, 4]; all predictions below 0 are set to 0 and all those above 4 are set to 4.

Unlike any of the other models used, FMs are capable of learning effectively from both categorical and real-valued features. We would like to maximize the amount of differentiating information which can be captured by the 2-way interaction factors of the model. For this purpose, when we have the choice between encoding a feature as categorical (one-hot encoded) or real-valued (single feature with ordinal value), we choose the categorical encoding. This allows unique 2-way interactions to be learned for each combination of categories for all categorical features. Among the techniques which can incorporate content features, the FM model is the only one which leverages MF techniques to learn these sparse 2-way interactions effectively. With this in mind, we do not include highly sparse features (such as the instructor IDs) as training data for the other models. This reduces computational overhead at an insignificant loss to performance.

6.2.1. FM Feature Selection: A Novel Importance Metric

In our experiments, we found that the FM model has particularly poor feature selection capabilities, unlike the L-1-regularized regression model and the decision-tree-based models. We introduce a new feature importance metric called Mean Absolute Deviation Importance (MADImp).

Inspired by the work of (Elbadrawy et al., 2015), MADImp is a method for computing the importance of each feature in any generalized linear model (GLM). Elbadrawy et. al. compute importance for a model term as the proportion of the overall prediction accounted for by that term, normalized over all records. This is greatly simplified by the non-negativity constraints enforced in the particular linear model used by those authors. Since not all GLMs have non-negativity constraints, we cannot use simple averaging. Instead, we use a quantity computed from the mean absolute deviation from a global intercept term.

For each dyad, the active features (non-zero) are involved in one or more additive interaction terms which move the prediction away from the global intercept w_0. To measure importance, we sum the absolute values of these additive terms for a particular feature and divide by the total absolute deviations caused by all features. Importance is measured as the proportion of the absolute deviation from w_0 accounted for by a particular feature. We average this over all records – hence the name Mean Absolute Deviation Importance (MADImp). For a practical example that solidifies these ideas, see Appendix III: MADImp Example.

We now define MADImp more formally. Throughout this section we use d as shorthand for some dyad indices i, j. The importance of each feature for a single dyad is calculated as:

$$I(\boldsymbol{X}_{d,f}) = \frac{\sigma_1(\boldsymbol{X}_{d,f}) + \sigma_2(\boldsymbol{X}_{d,f})}{T_d}, \tag{11}$$

where $\sigma_1(\boldsymbol{X}_{d,f})$ is the deviation from 1-way interactions of feature f in dyad d, $\sigma_2(\boldsymbol{X}_{d,f})$ is the deviation from 2-way interactions of feature f in dyad d, and T_d is the total deviation from the global intercept w_0 in the estimation for dyad d.

$$\sigma_1(\boldsymbol{X}_{d,f}) = |w_f \boldsymbol{X}_{d,f}| \tag{12}$$

$$\sigma_2(\boldsymbol{X}_{d,f}) = \boldsymbol{X}_{d,f}^2 \sum_{f'=1, f' \neq f}^{p} \frac{|\boldsymbol{X}_{d,f} Z_{f,f'}|}{|\boldsymbol{X}_{d,f}| + |\boldsymbol{X}_{d,f'}|} \tag{13}$$

$$T_d = \sum_{f=1}^{p} |w_f \boldsymbol{X}_{d,f}| + \sum_{f=1}^{p} \sum_{f'=f+1}^{p} |\boldsymbol{X}_{d,f} \boldsymbol{X}_{d,f'} Z_{f,f'}| \tag{14}$$

The sum of the importance measures of all features for a dyad equals the total deviation for that dyad: $\sum_{f=1}^{p} I(\boldsymbol{X}_{d,f}) = T_d$. We can calculate the overall importance of a feature in our training dataset through:

$$I(\boldsymbol{X}_{*,f}) = \sum_{d=1}^{p} I(\boldsymbol{X}_{d,f}) \tag{15}$$

In general, to calculate MADImp for $\boldsymbol{X}_{*,f}$ for any GLM, group all terms $\boldsymbol{X}_{*,f}$ is involved in and take the absolute value. Do the same for all other $\boldsymbol{X}_{*,f}$. Normalize by the sum of all term deviations $\sum_f \boldsymbol{X}_{*,f}$. Finally, take the mean across all training records.

After training the FM model on all academic terms, we average these importance metrics across terms, weighting by the number of records predicted in each term. By applying this metric, we found that the student bias, course bias, and instructor bias were most informative, followed by the course discipline, major, student race, and instructor rank. The cohort, instructor class and tenure, student term, transfer indicator, and sex were found to be less important but still informative. Training the FM with only these features greatly improves the results when compared with training on all features. **RMSE dropped from 1.0587 to 0.7832 – an improvement of 26%**. This demonstrates that using MADImp for FM feature selection can greatly improve results and significantly reduce the amount of effort required when searching for the most suitable features. For the other models, we relied on their inherent feature selection capabilities.

6.3. Grade Prediction Results

The prediction results are broken down by non-cold-start vs. cold-start in Tables 2. We first discuss results from methods not incorporating content features. On non-cold-start records, we see the UR and GM methods perform worst, as expected. It is somewhat surprising to see SVD perform worse than the MoM method. This is likely due to overfitting, which is a common issue with SVD (Arkadiusz Paterek, 2007). While the post-processing is able to improve the SVD results, they are not nearly as good as those produced by the MoM method. This indicates the per-student and per-course biases are quite informative. The FM model outperforms all the others by a wide margin, indicating 2-way feature interactions play an important role in predicting performance. This also indicates that proper regularization noticeably improves the generalization of learned patterns for prediction in future terms.

All methods except the MoM method actually show improved results when predicting on cold-start records. Since the GM method improves, we can conclude the grades are simply less spread among cold-start dyads, making the task simpler than is usually the case. This is likely an artifact of the large number of transfer credits. These are all cold-start dyads since transfer students are all previously unseen in the dataset. Since only passing grades can transfer from other universities, these transfer credits shift the grade distribution upwards and make it more centered around the mean. The MoM method goes from being the best to being the worst. This indicates that relying on only one or two of the bias terms leads to over-confidence in predictions. In this setting, the FM model is relying on essentially the same information as the MoM method, but it is able to avoid overconfident learning patterns through the use of Bayesian complexity control (integrating out the regularization hyperparameters) for proper regularization.

We next discuss results for prediction with content features included (see Table 2). The FM model continues to produce the lowest-error predictions. The next best method is PLMR, followed by the RF. For non-cold start dyads, all of these methods outperform MoM, but only the FM with content features outperforms the FM without. The same thing cannot be said for

Table 2: Non-Cold Start vs. Cold-Start Prediction RMSE

Method	NCS RMSE	NCS MAE	CS RMSE	CS MAE
Without Content Features				
FM	**0.7792**	$\mathbf{0.56 \pm 0.54}$	**0.7980**	$\mathbf{0.61 \pm 0.51}$
MoM	0.8643	0.64 ± 0.58	1.4788	1.30 ± 0.70
SVD kNN	0.9249	0.68 ± 0.62	0.8123	0.67 ± 0.46
SVD	0.9370	0.69 ± 0.63	0.8095	0.67 ± 0.46
GM	0.9448	0.72 ± 0.61	0.8144	0.68 ± 0.45
UR	1.8667	1.54 ± 1.06	1.8977	1.57 ± 1.07
With Content Features				
FM	**0.7423**	$\mathbf{0.52 \pm 0.53}$	0.8111	0.61 ± 0.53
PLMR	0.7886	0.57 ± 0.55	1.0207	0.75 ± 0.70
RF	0.7936	0.58 ± 0.54	**0.7475**	$\mathbf{0.59 \pm 0.46}$
kNN	0.8061	0.59 ± 0.55	0.8101	0.65 ± 0.49
SGD	0.8207	0.60 ± 0.56	1.0297	0.75 ± 0.71

the FM on cold-start records. In this setting, the RF model is the clear winner. The FM model performs just slightly worse than the kNN regressor, and the SGD model performs worst.

Of these models, the SGD regressor is the only one limited to a linear fit, since it uses only 1-way interaction terms. Its relatively poor performance indicates there are definite non-linear trends correlated with performance outcomes. The kNN model performs slightly worse than the RF model on non-cold-start records but does not degrade significantly on cold-start records. This shows that kNN is able to capture somewhat more general trends, which is expected of a clustering technique that infers patterns from shared membership in a subpopulation. It may be that kNN fails to capture the trends present in a large set of historical data because it breaks the data up into smaller subsets and therefore loses out on more general trends that emerge over time. It may also be the case that it fails to identify patterns in ordinal data that would be further delineated by examining a larger population.

It is particularly interesting to notice that the FM with content features performs worse than the FM without on cold-start prediction. This indicates the FM model is learning patterns from prior terms that no longer hold in the current (prediction) term. It is able to capture 2-way interactions between features while other methods are not. So these patterns seem to be shifting significantly as new students enroll. This problem, where the test distribution differs significantly from the training distribution, is commonly known as *covariate shift* (Shimodaira, 2000). Uncovering inability to deal with covariate shift as a limiting factor in one of our best methods allows us to target future research at overcoming this problem. We discuss these implications in more detail in Section 7.

Table 3 provides an alternative breakdown of the prediction results, comparing native students to transfer students. We observe significantly better prediction results for transfer students than for native students and note the gap is narrowed when incorporating content features. This observation is another reflection of the reduced spread of the transfer grade distribution as compared to native grades. Notice that the best performing methods closely align with the non-cold-start vs. cold-start results from Table 2: FM performs best on native students while RF performs best on transfer students. This largely reflects the proportion of cold-start dyads

Table 3: Native vs. Transfer Prediction RMSE

Method	Native RMSE	Native MAE	Transfer RMSE	Transfer MAE
Without Content Features				
FM	**0.8054**	$\mathbf{0.58 \pm 0.56}$	**0.7805**	$\mathbf{0.62 \pm 0.48}$
SVD-kNN	0.9343	0.69 ± 0.63	0.8345	0.67 ± 0.50
SVD	0.9519	0.70 ± 0.65	0.8325	0.67 ± 0.49
GM	0.9576	0.72 ± 0.63	0.8391	0.68 ± 0.49
MoM	1.0076	0.77 ± 0.65	1.3064	1.08 ± 0.74
UR	1.8679	1.54 ± 1.06	1.8899	1.56 ± 1.07
With Content Features				
FM	**0.7879**	$\mathbf{0.55 \pm 0.57}$	0.7807	$\mathbf{0.58 \pm 0.52}$
RF	0.8062	0.59 ± 0.55	**0.7515**	0.59 ± 0.47
kNN	0.8346	0.61 ± 0.57	0.7936	0.62 ± 0.49
PLMR	0.8860	0.62 ± 0.64	0.9258	0.68 ± 0.63
SGD	0.9137	0.65 ± 0.66	0.9402	0.69 ± 0.64

from each sample: only 7.37% of native dyads are cold-start while 42.62% of transfer dyads are cold-start. The greater diversity of grades among the native students makes the content features much more important for accurate predictions.

Table 4 lays out the results for the top three methods (FM, PLMR, and RF) with a more detailed cold-start vs. non-cold-start error breakdown. The FM model is outperformed when student information is absent; in these cases, the RF model performs best. PLMR only performs well on non-cold-start records. These results indicate that we can improve our next-term grade prediction system by swapping out RF for FM whenever we are lacking prior student information. Doing so gives an overall RMSE of 0.7443 compared to 0.7709 for FM and 0.7775 for RF. This demonstrates that such a hybrid is a viable solution to overcoming the cold-start problems suffered by FMs.

6.4. Feature Importance

It is useful to have some notion of which features are most informative for next-term grade prediction. This understanding can eventually help us to uncover the relationships between student performance and the various predictor variables available to us. These features differ between the FM, RF, and PLMR models. While FMs can leverage sparse categorical features effectively via matrix factorization, decision trees are typically unable to discover useful patterns in such data. The PLMR model can leverage 1-way interactions from categorical features, but they must rely on real-valued features to learn useful trends in the absence of the 2-way interactions of the FM model.

We start by examining the importance of features in the FM model. Figure 1 shows the importance of features broken down by 1-way and 2-way interactions. The top 3 features make up 97.3% of the importance. These are the student bias (sid), course bias (cid), and instructor bias (iid). Roughly two thirds of the importance comes from 2-way interactions, and the other third comes from the 1-way interactions. Of the non-bias terms, the course discipline (cdisc) and major are the only features which have any notable importance, accounting for 1.1% and 0.8%. This indicates that these high-level interactions can account for the majority of variance

Table 4: Cold-start vs. Non-Cold-start Error

Group	Dyad %	Method	RMSE	MAE
NCS	48.60	**FM**	**0.7423**	**0.5187** $\pm$ **0.5310**
		PLMR	0.7890	0.5635 $\pm$ 0.5522
		RF	0.7936	0.5837 $\pm$ 0.5377
CSS	42.31	**RF**	**0.7381**	**0.5867** $\pm$ **0.4478**
		FM	0.8112	0.6114 $\pm$ 0.5331
		PLMR	0.9917	0.7321 $\pm$ 0.6689
CSC	1.75	**FM**	**0.7456**	**0.5293** $\pm$ **0.5252**
		RF	0.7776	0.5695 $\pm$ 0.5295
		PLMR	1.1771	0.7489 $\pm$ 0.9081
CSB	4.55	**RF**	**0.8203**	**0.6603** $\pm$ **0.4867**
		FM	0.8337	0.6614 $\pm$ 0.5075
		PLMR	1.2060	0.8829 $\pm$ 0.8215

CS = cold-start, NCS = non-CS, CSS = CS student-only,
CSC = CS course-only, CSB = CS both

in predictions.

The PLMR model does not learn 2-way interactions between features, so there is much more variety in the importance metrics. Figure 2 shows the evolution of feature importance across terms, excluding the summer terms (which have very few records). The individual lines represent the percent of the overall importance accounted for by a particular feature in a particular term. Each segment/chunk in a line represents 2% of the overall importance. The X-axis represents the terms, moving from Fall 2009 to Spring 2014.

We see a clear shift from the first term to the second and less drastic shifts moving forward from there. By the last term, the instructor bias, classification, rank, and tenure status, the number of terms a student has been enrolled, and the course discipline emerge as the most important features. We see that last-term GPA features gain in importance as more data is acquired. In contrast, the number of students enrolled in a course during the current term and in prior terms both decline in importance.

Figure 3 shows the overall PLMR feature importances for the transfer vs. the non-transfer data. These were obtained by training separate models on the transfer data and the non-transfer data. The most noticeable difference is the massive importance of the course bias (accounting for nearly 60% of overall bias) in the non-transfer data compared to the complete lack of course bias importance in the transfer data. This is most likely due to the mapping from transfer credits to their GMU equivalents. A variety of courses from many different universities can be mapped to the same GMU course. So the mapping process dilutes the usefulness of this feature. This also explains the lack of importance of the course discipline in the transfer data. For the transfer population, the lack of useful course bias terms causes the PLMR model to learn larger weights for the other features. Discounting this general trend, there are still marked differences in importance for the number of terms a student has been enrolled, instructor classification, and the student and instructor bias terms. Finally, we note that the importance of the iid feature reflects the institution id in all dyads obtained from transfer records. Since this is one of the more useful features learned for the transfer performance prediction, it seems substitution of instructor id with instituion id is a reasonable preprocessing step when predicting performance in these cases.

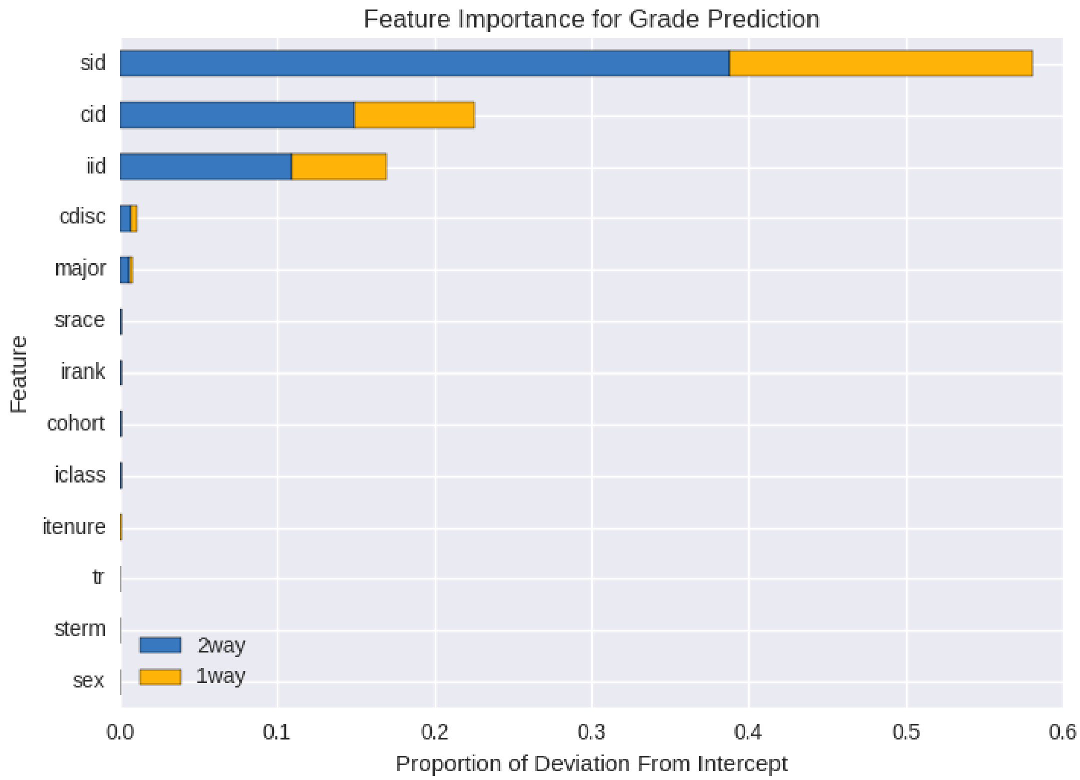


Figure 1: FM: Overall MADImp Feature Importance

The RF model performs binary splits on the input data using one particular feature at a time. The Gini Coefficient is used to determine the most informative splits at each level of the tree. Gini Importance (GI) is computed as the normalized total reduction in mean squared error (MSE) brought by each feature over all splits. Due to the normalization, this can also be thought of as the percent of the MSE reduction achieved. We compute GI for each term, including predictions for cold-start records, then average them to get the final GI proportions. The top 10 most important features of the RF in order of highest to lowest GI are shown in Table 5.

For the RF model, the most informative are the grade, credit hour, and course enrollment features. Together, these features characterize how competent a student generally is and how difficult and well-established a course generally is. The prior GPA gives some notion of historical competency, but this weighs in less heavily than more recent college grades. The zip code is the one demographic feature that shows up here, indicating that demographics are important when no student-specific bias terms can be learned. The instructor classification shows up again here as well.

It is particularly interesting to note that the instructor features were important for all three methods. This indicates that the particular instructor and his or her rank, classification, and tenure status have detectable effects on student grades in particular courses. To our knowledge, no previous studies have explored the effect of instructor biases or interactions for grade prediction. Having clearly characterized their importance here, we hope further research in this field can capitalize on the information present in such features to improve educational data mining applications.

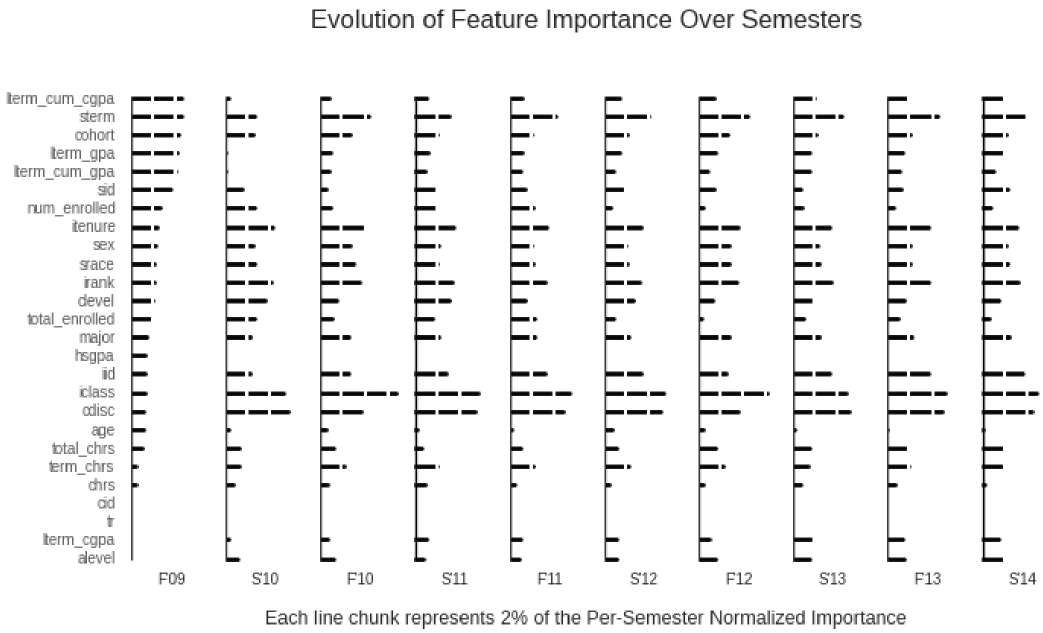


Figure 2: PLMR: Feature Importance Evolution for All Data

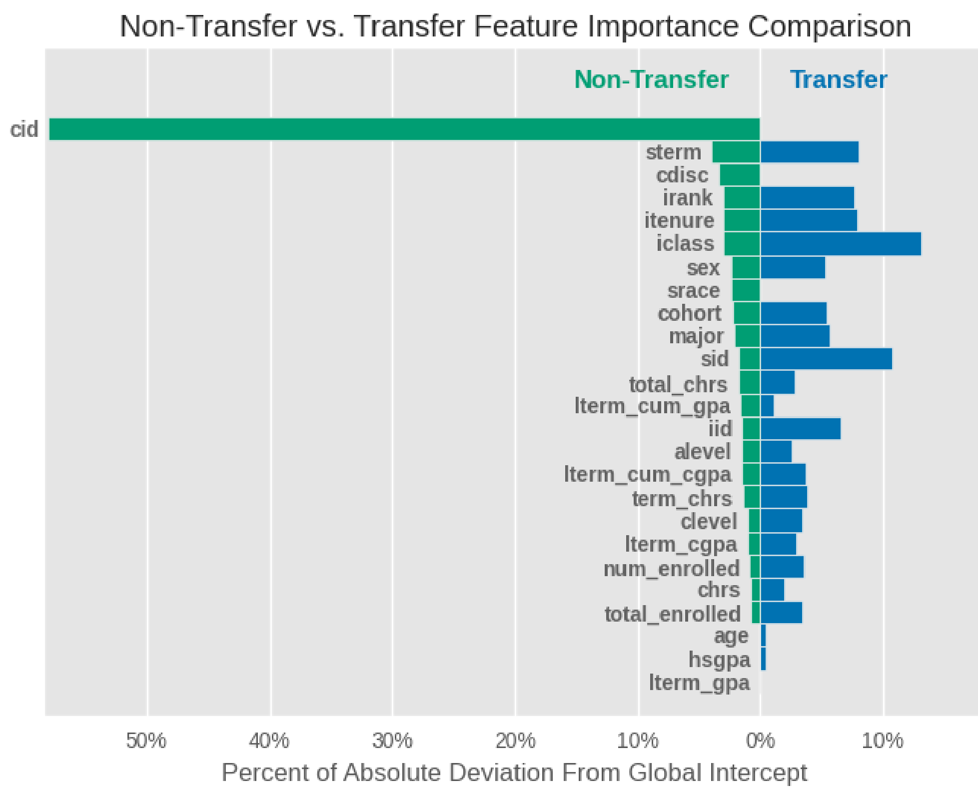


Figure 3: PLMR: Feature Importance Comparison

Table 5: RF: Top 10 Features by Gini Importance (GI)

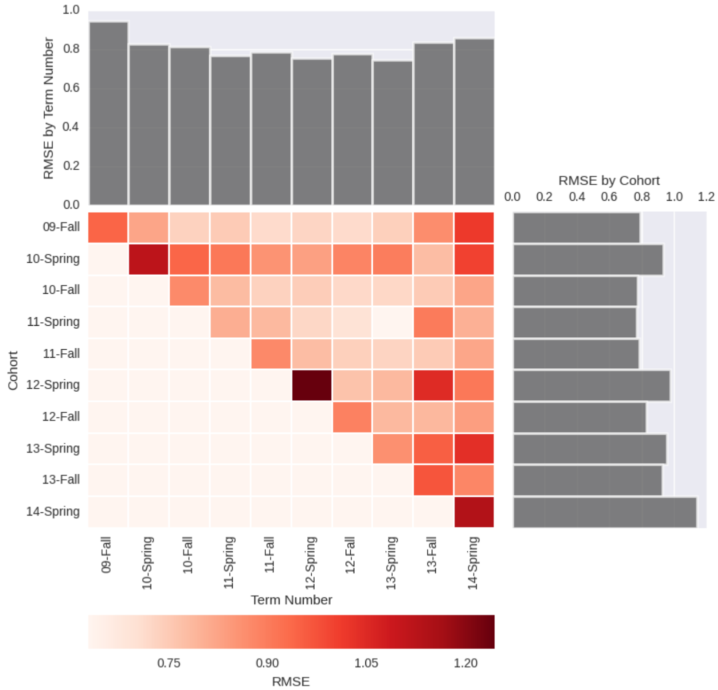
Feature	GI	Description
lterm_gpa	0.4164	Student's last-term GPA
lterm_cum_cgpa	0.2484	Course's last-term cumulative GPA
lterm_cum_gpa	0.0645	Student's last-term cumulative GPA
prior_gpa	0.0539	Student's high school or transfer GPA
lterm_cgpa	0.0349	Course's last-term GPA
total_chrs	0.0197	Credit hours attempted by student
term_chrs	0.0175	Credit hours student is taking this term
total_enrolled	0.0170	Total historical course enrollment
zip	0.0153	Student zip code
iclass	0.0147	Instructor classification

6.5. Error Analysis

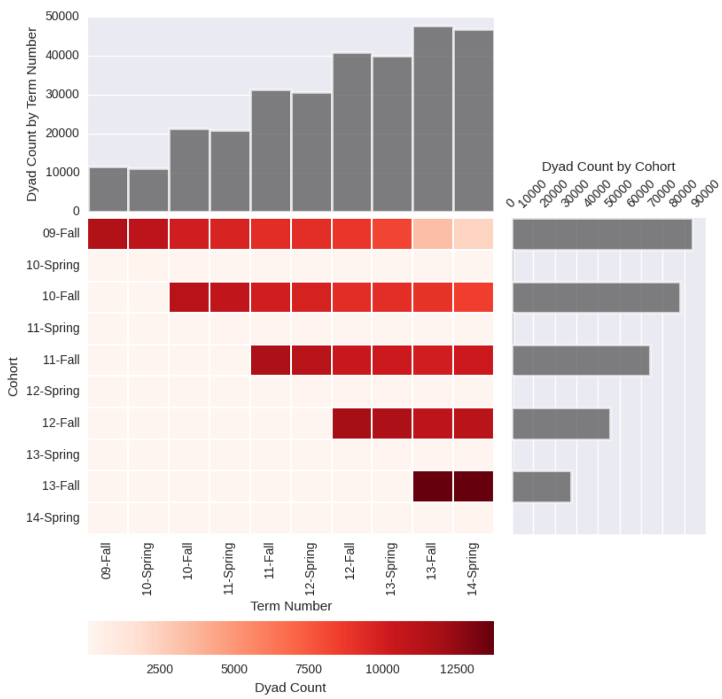
Fig. 4a lays out the distribution of error for the FM model in terms of RMSE in a cohort by term number matrix for the non-transfer students. The cohort is the term the student was admitted to the university, while the term number is the objective chronological ordering of the academic terms in the dataset. Each cell represents the aggregate RMSE for all grade predictions for the students admitted in a particular cohort taking courses in a particular academic term. The bar chart on top of the heatmap shows the per-term RMSE, and the bar chart on the right shows the per-cohort RMSE. These errors are from FM with the best settings using content features and predicting for all records, including cold-start. Note that we have excluded Summer terms, which account for only 2.67% of the predictions. This clarifies the trends observed. Also note that Spring cohorts account for only 1.10% of the total predictions after removing summer terms. Hence they have much more variability than the Fall terms. Fig. 4b shows a heatmap of the dyad counts for each of the cohort/term cells.

Looking down the diagonal, we see that predictions for each cohort for the first term are generally poorer than predictions for subsequent terms. This reflects the increased difficulty of cold-start predictions, since these are all cold-start students and may also be cold-start courses (new courses). Following each row from left to right, we expect the predictions to decrease in error as we accumulate more information about the students in this cohort. With the exception of the Fall of 2013 and Spring of 2014, we do see this trend. So in general, our predictions improve as we accumulate more historical grade data.

We hypothesize that the degradation of performance moving into the last two terms is indicative of covariate shift in the dataset. The first large cohort in our dataset enrolled in Fall 2009. So those students are beginning to graduate in significant numbers in 2013. Nearly 1000 students graduated in the Spring of 2013, followed by another 500 or so in the summer and Fall. Then another approximately 1200 students graduate in the Spring of 2014. These transitions represent a significant shift in the characteristics of the dataset and also remove from the dataset our most well-known students. These effects taken together seem to largely explain the performance degradation moving into these terms and provide further evidence for covariate shift in such university data.



(a) Error breakdown for FM predictions, excluding summer terms.



(b) Dyad counts for each cell in error breakdown.

Figure 4: Cohort by term number error visualization for non-transfer students.

7. Discussion

7.1. Predictive Performance

We found that a FM-RF hybrid is an effective method of overcoming the cold-start limitations of FMs and is the most effective method in general. However, we also note that more sophisticated methods have been developed to improve performance of MF-based methods on cold-start prediction tasks. One notable example is the attribute-to-latent-feature mapping developed by (Gantner et al., 2010). Such an approach might produce better results and/or be computationally cheaper than the FM-RF hybrid.

We also uncovered covariate shift as a likely source of error when using FMs for cold-start prediction. When the test distribution differs from the training distribution, the FM model is liable to learn overconfident 2-way interactions that reduce its ability to generalize. The RF model does not seem to suffer from this problem, but it is also clearly unable to capture 2-way interactions. Discovering new ways to handle covariate shift while learning 2-way interactions might be another viable way to overcome cold-start issues in MF-based methods.

7.2. Beneficial Applications

Using our system, we can predict student grades in the next enrollment term much more accurately than a random guessing strategy or the regression models commonly used in the higher education literature would. This information can be used to aid students, educators, and advisors in a variety of ways. **For students**, we can incorporate this information into a degree planning system. Such systems already have ways of determining which courses meet which degree requirements (Parameswaran et al., 2011). Given several sets of possible course selections for a semester, our system can be used to maximize the expected grade. We could also provide students with a personalized difficulty rating for each course based on their expected grade. This would help students prioritize studies for each course and prepare properly for particularly challenging courses.

For educators, knowledge of which students have the lowest expected grades could provide opportunities to increase detection of at-risk students. While students can seek help and have access to many resources, research shows that at-risk students perform better if instructors are proactive in identifying and reaching out to them (Grayson et al., 1998). A great deal of research has gone into identifying characteristics of effective interventions based on information provided by learning analytics (Arnold and Pistilli, 2012). The Open Source Analytics Initiative (OSAI) conducted a survey and analysis of this body of research, identifying several key characteristics (Jayaprakash et al., 2014). Interventions that increase student communication with instructors, connect students to available university services, and provide students with an *accurate assessment of how they are currently performing* can be effective at increasing both retention and academic performance. Our grade prediction system can be used as a component of an early-warning system that can equip instructors to provide interventions with these crucial characteristics.

The information provided by our system would also be invaluable **for advisors**. Anticipating student performance and recommending the best plan of action is a critical component of advisor responsibilities (Aiken-Wisniewski et al., 2015). Any additional information that helps them personalize their advice to each student could potentially help thousands of students. For instance, imagine a particular course is known to be a challenging bottleneck course and another course is known to be particularly easy in general. Without additional information, an advisor might always recommend students take these two courses in the same semester. However, some students might have trouble with exactly the type of material taught or exactly the type of teaching style used in the "easy" course. Our system could provide advisors with this type of personalized information.

8. Conclusions and Future Work

Motivated by the need for institutions to retain students, ensure timely graduation, and ensure students are well-trained and workforce-ready in their field of study, we have leveraged state-of-the-art recommender systems techniques to develop a system for accurate next-term student grade prediction. After experimenting with a wide variety of regression and factorization models, we determined that a hybrid of the Random Forest model and the MF-based Factorization Machine is best-suited to this task. Key to the success of this hybrid model is the application of MADImp to FM feature selection. Using this hybrid, we can predict grades for both new and returning students and for both new and existing courses. While this system significantly outperforms random guess predictions, and noticeably outperforms the other regression models tested, it still has limitations we seek to address in future works. The main limitations are poorer performance for grades at the extremes and difficulty dealing with covariate shift as students graduate and new students enroll.

Of the methods surveyed, tensor factorization, multi-relational matrix factorization, and custom tree-based methods seem most promising. The first two methods are inspired by similar work in the ITS community. Both of these methods hold promise for effectively leveraging the valuable instructor features identified in this study. The idea of using more customized tree-based methods is motivated by the desire for interpretable decision-making rules. Easily understandable decision rules can speed student, educator, and advisor adoption of new practices motivated by our findings. Additional feature engineering and custom methods of constructing and pruning decision trees seem promising areas for further work.

As in other MF applications, we observed the cold-start problem is a limiting factor and complementary methods were required to overcome it. We overcame this problem by combining FMs with RFs to compose a hybrid model. We have also discussed using attribute-to-latent-feature mappings and identifying means of overcoming covariate shift while learning 2-way interactions as two possible augmentations to help FMs overcome the cold-start problem. Further research in this area could greatly benefit many fields where MF-based methods are applicable.

Once we have improved predictive performance for both failing and passing grades, we plan to deploy this system for live use as a component of a degree planner and an early-warning system for students, instructors, and advisors. Once live, we would perform A/B testing to better understand its performance, the effect it has on the decision-making of its various users, and its ability to improve student retention and learning success outcomes.

Acknowledgment

This research was funded by NSF IIS grant 1447489. The data for this study was made available through a collaborative effort spearheaded by Office of Institutional Research and Reporting and we would like to acknowledge Kris Smith, Kathryn Zora, Angela Detlev, Eve Dauer, Kathy Zimmerman, Joe Balducci, and Judy Lou at GMU.

REFERENCES

2014. Gmu facts 2014-2015 facts and figures.

ADAMIC, L. A., LUKOSE, R. M., PUNIYANI, A. R., AND HUBERMAN, B. A. 2001. Search in power-law networks. *Physical Review E 64,* 4.

ADOMAVICIUS, G. AND TUZHILIN, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng. 17,* 6, 734–749.

AIKEN-WISNIEWSKI, S. A., JOHNSON, A., LARSON, J., AND BARKEMEYER, J. 2015. A Preliminary Report of Advisor Perceptions of Advising and of a Profession. *NACADA Journal 35,* 2 (Nov.), 60–70.

ALSUMAIT, L., WANG, P., DOMENICONI, C., AND BARBARA, D. 2010. Embedding semantics in LDA topic models. In *Text Mining*, M. W. Berry and J. Kogan, Eds. John Wiley & Sons, Ltd, 183–204.

ARKADIUSZ PATEREK. 2007. Improving regularized singular value decomposition for collaborative filtering. *KDD*.

ARNOLD, K. E. AND PISTILLI, M. D. 2012. Course Signals at Purdue: Using Learning Analytics to Increase Student Success. In *Proceedings of the 2Nd International Conference on Learning Analytics and Knowledge*. LAK '12. ACM, New York, NY, USA, 267–270.

AUD, S. AND WILKINSON-FLICKER, S. 2013. *The Condition of Education 2013*. Number NCES 2013-037. U.S. Department of Education, National Center for Education Statistics.

BAKER, R. S., CORBETT, A. T., AND ALEVEN, V. 2008. Improving contextual models of guessing and slipping with a truncated training set. *Human-Computer Interaction Institute*, 17.

BARNES, T. AND STAMPER, J. 2008. Toward automatic hint generation for logic proof tutoring using historical student data. In *Proc. of the 9th International Conference on Intelligent Tutoring Systems* (2008). 373–382.

BAYER, I. 2015. fastFM: A library for factorization machines. *arXiv:1505.00641v2*.

BELL, R. M., KOREN, Y., AND VOLINSKY, C. 2007. The BellKor solution to the netflix prize. *AT&T Labs Research.*

BREIMAN, L. 2001. Random forests. *Machine Learning 45,* 1, 5–32.

CORBETT, A. T. AND ANDERSON, J. R. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction 4,* 4, 253–278.

COUNCIL, N. R. 2001. *Building a Workforce for the Information Economy*. National Academies Press.

DESMARAIS, M. C. AND BAKER, R. S. J. D. 2011. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction 22,* 1, 9–38.

DRACHSLER, H., VERBERT, K., SANTOS, O. C., AND MANOUSELIS, N. 2015. Panorama of Recommender Systems to Support Learning. In *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. Springer US, Boston, MA, 421–451.

ELBADRAWY, A., STUDHAM, R. S., AND GEORGE KARYPIS. 2015. Personalized multi-regression models for predicting students' performance in course activities. *LAK, '15*.

EVANS, B. J., BAKER, R. B., AND DEE, T. S. 2016. Persistence Patterns in Massive Open Online Courses (MOOCs). *Journal of Higher Education 87,* 2 (Apr.), 206–242.

FREUDENTHALER, C., SCHMIDT-THIEME, L., AND RENDLE, S. 2009. Factorization Machines Factorized Polynomial Regression Models.

GANTNER, Z., DRUMOND, L., FREUDENTHALER, C., RENDLE, S., AND SCHMIDT-THIEME, L. 2010. Learning attribute-to-feature mappings for cold-start recommendations. In *Proc. of the 2010 IEEE International Conference on Data Mining* (2010). ICDM '10. IEEE Computer Society, 176–185.

GRACE KENA, LAUREN MUSU-GILLETTE, JENNIFER ROBINSON, XIAOLEI WANG, AMY RATHBUN, JIJUN ZHANG SIDNEY WILKINSON-FLICKER, AMY BARMER, AND ERIN DUNLOP VELEZ. 2015. The Condition of Education 2015. Tech. Rep. NCES 2015-144, U.S. Department of Education, National Center for Education Statistics, Washington, DC. May.

GRAYSON, A., MILLER, H., AND CLARKE, D. D. 1998. Identifying barriers to help-seeking: a qualitative analysis of students' preparedness to seek help from tutors. *British Journal of Guidance and Counselling 26,* 2, 237–253.

HUETE, J. F., FERNNDEZ-LUNA, J. M., DE CAMPOS, L. M., AND RUEDA-MORALES, M. A. 2012. Using past-prediction accuracy in recommender systems. *Information Sciences 199*, 78–92.

JAYAPRAKASH, S. M., MOODY, E. W., LAURA, E. J., REGAN, J. R., AND BARON, J. D. 2014. Early alert of academically at-risk students: An open source analytics initiative. *1,* 1, 6–47.

KANAGAL, B., AHMED, A., PANDEY, S., JOSIFOVSKI, V., YUAN, J., AND GARCIA-PUEYO, L. 2012. Supercharging recommender systems using taxonomies for learning user purchase behavior. *Proc. VLDB Endow. 5,* 10, 956–967.

KEVIN CAREY. 2005. Choosing to improve: Voices from colleges and universities with better graduation rates.

KOREN, Y. AND BELL, R. 2011. Advances in collaborative filtering. In *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 145–186.

LINDEN, G., SMITH, B., AND YORK, J. 2003. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing 7,* 1, 76–80.

LIYANAGUNAWARDENA, T. R., ADAMS, A. A., AND WILLIAMS, S. A. 2013. MOOCs: A systematic study of the published literature 2008-2012. *The International Review of Research in Open and Distributed Learning 14,* 3 (June), 202–227.

NEDUNGADI, P. AND SMRUTHY, T. K. 2016. Personalized multi-relational matrix factorization model for predicting student performance. In *Intelligent Systems Technologies and Applications*, S. Berretti, S. M. Thampi, and P. R. Srivastava, Eds. Vol. 384. Springer International Publishing, 163–172.

OGNJANOVIC, I., GASEVIC, D., AND DAWSON, S. 2016. Using institutional data to predict student course selections in higher education. *The Internet and Higher Education 29*, 49–62.

PARAMESWARAN, A., VENETIS, P., AND GARCIA-MOLINA, H. 2011. Recommendation systems with complex constraints: A course recommendation perspective. *ACM Transactions on Information Systems (TOIS) 29,* 4, 20.

PARDOS, Z. A., WANG, Q. Y., AND TRIVEDI, S. 2012. The real world significance of performance prediction. *International Educational Data Mining Society*.

PEÑA-AYALA, A. 2014. Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications 41,* 4, 1432–1462.

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A. E. A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M., AND DUCHESNAY, E. 2011. Scikit-learn: Machine learning in Python. *JMLR 12*, 2825–2830.

RENDLE, S. 2012. Factorization machines with libFM. *ACM TIST 3,* 3.

RHODES, N., UNG, M., ZUNDEL, A., HEROLD, J., AND STAHOVICH, T. 2013. Using a lexical analysis of student's self-explanation to predict course performance. In *Educational Data Mining 2013* (2013).

ROMERO, C., LÓPEZ, M., LUNA, J., AND VENTURA, S. 2013. Predicting students' final performance from participation in on-line discussion forums. *Computers & Education 68*, 458–472.

ROMERO, C. AND VENTURA, S. 2010. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 40,* 6 (Nov.), 601–618.

ROMERO, C., VENTURA, S., AND GARCÍA, E. 2008. Data mining in course management systems: Moodle case study and tutorial. *Computers & Education 51,* 1, 368–384.

SALAKHUTDINOV, R., MNIH, A., AND HINTON, G. 2007. Restricted boltzmann machines for collaborative filtering. In *Proc. of the 24th ICML*. ACM, 791–798.

SHAN, H. AND BANERJEE, A. 2010. Generalized probabilistic matrix factorizations for collaborative filtering. In *ICDM '10* (2010-12). 1025–1030.

SHANI, G., HECKERMAN, D., AND BRAFMAN, R. I. 2005. An MDP-based recommender system. *JMLR 6*, 1265–1295.

SHIMODAIRA, H. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference 90,* 2, 227–244.

SOROUR, S. E., MINE, T., GODA, K., AND HIROKAWA, S. 2015. A predictive model to evaluate student performance. *Journal of Information Processing 23,* 2, 192–201.

SWEENEY, M., LESTER, J., AND RANGWALA, H. 2015. Next-term student grade prediction. In *2015 IEEE International Conference on Big Data (Big Data)*. 970–975.

THAI-NGHE, N., DRUMOND, L., HORVÁTH, T., NANOPOULOS, A., AND SCHMIDT-THIEME, L. 2011. Matrix and tensor factorization for predicting student performance. In *CSEDU (1)* (2011). 69–78.

THAI-NGHE, N., DRUMOND, L., HORVÁTH, T., AND SCHMIDT-THIEME, L. 2011. Multi-relational factorization models for predicting student performance. In *KDD Workshop on Knowledge Discovery in Educational Data (KDDinED)* (2011).

THAI-NGHE, N., DRUMOND, L., HORVÁTH, T., AND SCHMIDT-THIEME, L. 2012. Using factorization machines for student modeling. In *UMAP Workshops* (2012).

TINTO, V. 2006. Research and practice of student retention: what next? *Journal of College Student Retention: Research, Theory & Practice 8,* 1, 1–19.

TINTO, V. 2012. *Completing College: Rethinking Institutional Action*. University of Chicago Press.

XU, Y. AND MOSTOW, J. 2012. *Comparison of Methods to Trace Multiple Subskills: Is LR-DBN Best?* International Educational Data Mining Society.

XU, Y. AND MOSTOW, J. 2015. A unified 5-dimensional framework for student models. *International Educational Data Mining Society*.

Appendix I: Feature Descriptions

This section provides detailed descriptions of each feature.

8.1. Student Features

- *sid*: Unique identifier of the student. When used in training data, the student ID is one-hot encoded to learn student bias terms.
- *grdpts*: [0, 4] grade the student has obtained for a particular course.
- *major*: Declared major during current term. Students may change majors, so the same student may take courses with a different declared major at different times.
- *race*: Self-reported race of student; may be unspecified.
- *sex*: Self-reported gender; may be unspecified.
- *age*: Age determined from birth date in admissions records.
- *zip*: Zip code, or postal code for students from outside the US.
- *sat*: 1600-scale SAT score, if available. The SAT is a standardized test commonly used for college admissions in the United States. The reading and math sections are often considered separately from the writing. Both are scored from 200-800, giving a max score of 1600.
- *hs*: High school CEEB code. These are standardized ID numbers given to high schools, colleges, and universities by the Educational Testing Service (ETS). They are used to uniquely identify high schools in our dataset.
- *hsgpa*: High school GPA. For transfer students, this feature contains the GPA from the institution the student is transferring from.
- *lterm_gpa*: Grade Point Average (GPA) from the previous term.
- *lterm_cum_gpa*: Cumulative GPA as of the previous term. The cumulative GPA is an average of course grades weighted by the course credit hours.
- *term_chrs*: Number of credit hours the student is enrolled in during the current term.
- *total_chrs*: Number of credit hours student has taken (not passed) up to the previous term.
- *alevel*: Academic level of the student. Obtained by binning total_chrs: [0,30)=0, [30,60)=1, [60,90)=2, [90,120]=3, (120+)=4. This is a measure of experience, rather than of progress in the program. It reflects total credit hours attempted, not just those from courses passed.
- *sterm*: Chronological numbering of terms relative to this student. The student's first term is 0, his second is 1, and so on.

8.2. Course Features

- *cid*: Unique identifier of the course. When used in training data, the student ID is one-hot encoded to learn course bias terms.
- *cdisc*: Course discipline.
- *chrs*: Number of credit hours this course is worth. This is a measure of expected work required by the course and is used to determine the importance of the course in GPA calculations.
- *clevel*: Course level [1, 7].
- *termnum*: Number of the term this course was offered in. These are relative to the dataset only. Term 0 is the first term for which we have data, and they are numbered chronologically onwards from there.

- *num_enrolled*: Number of students enrolled in this course for the current term, across all sections.
- *total_enrolled*: Total number of students enrolled in this course since its first offering, including the current term.
- *lterm_cgpa*: The aggregate [0, 4] GPA of all students who took this course during the previous term.
- *lterm_cum_cgpa*: The aggregate [0, 4] GPA of all students who have ever taken this class up to the previous term.

8.3. Instructor Features

- *iid*: Unique identifier of the instructor. When used in training data, the student ID is one-hot encoded to learn instructor bias terms.
- *iclass*: Classification (Adjunct, Full time, Part time, GRA, GTA).
- *irank*: Rank (Instructor, Assistant Professor, Associate Professor, Eminent Scholar, University Professor).
- *itenure*: Tenure status (Term, Tenure-track, Tenured).

Appendix II: Model Parameter Settings

- *Factorization Machine*: k = rank = 8, i = number of iterations = 200, s = initial standard deviation = 0.2
- *Personalized Linear Multiple Regression*: k = number of models = 4, lw = regularization on P and W = 0.01, lb = λ_B = 0.5, lr = SGD learning rate = 0.001
- *Random Forest*: n = number of trees = 100, m = max depth = 10
- *Boosted Decision Trees*: n = number of trees = 100, m = max depth = 11
- *Stochastic Gradient Descent (SGD) Regression*: lr = learning rate = 0.001, r = regularization term = 0.001, i = number of iterations = 15
- *Ordinary Least Squares (OLS) Regression*: no parameters
- *k-Nearest Neighbors (kNN)*: k = number of neighbors = 20
- *Decision Tree*: m = max depth = 4

Appendix III: MADImp Example

Let us consider a particular example to better understand the idea of Mean Absolute Deviation Importance (MADImp). We proceed with the FM model but note again that MADImp can be used with any generalized linear model (GLM).

Assume we have three features: a user ID, an item ID, and a season (Spring, Summer, Fall, Winter). We one-hot encode all features, so for a particular record we have three features with a value of 1 and all the rest have values of 0. Now the goal is to predict the response a particular user gives to a particular item during a particular season. Let us now fixate upon a single user. This user generally tends to give all items a slightly positive response. The responses are more positive in the Spring and Fall, and less positive in the Summer and Winter. Next we fixate upon a single item. This item usually generates very positive responses, with the positivity spiking in the Summer. The other seasons have no effect on users' responses to the item. The user has no particular preferences for or against the item. We also observe the overall pattern of responses is slightly positive and the range of ratings is from 0 to 4. Finally, we do not observe any general seasonality effects. We only find seasonal preferences of and for particular users and items.

Given this setting, we can now assign hypothetical values to our parameters. We have 10 users and 20 items, and there are 4 seasons. So after one-hot encoding, we have a total of 34 features. Let our user be feature 1, our item be feature 11, and the seasons are then the following feature numbers: Spring = 31, Summer = 32, Fall = 33, Winter = 34. As is typical with regression models, we set feature 0 to 1 for all records to learn the global intercept term.

- $w_0 = 0.5$, a positive global intercept term.
- $w_1 = 0.5$, a positive response trend for this user.
- $w_{11} = 2.0$, a very positive response trend for this item.
- $Z_{1,31} = Z_{1,33} = 0.2$, this user has slightly more positive responses in Spring and Fall.
- $Z_{1,32} = Z_{1,34} = -0.2$, this user has slightly less positive responses in Summer and Winter.
- $Z_{11,32} = 0.2$, this item has slightly more positive responses in the Summer.

With these parameter values, a prediction for this user-item combo in the Summer would be calculated as:

$$\begin{aligned}\hat{y}(X_d) &= w_0 + w_1 + w_{11} + Z_{1,32} + Z_{11,32} \\ &= 0.5 + 0.5 + 2.0 - 0.2 + 0.2 \\ &= 3.0\end{aligned}$$

Recall that we have three features set to 1 for each dyad feature vector X_d. So we calculate the importance of these three features using (11).

$$\begin{aligned}T_d &= |w_1| + |w_{11}| + |Z_{1,32}| + |Z_{11,32}| \\ &= |0.5| + |2.0| + |-0.2| + |0.2| = 2.9\end{aligned}$$

$$\begin{aligned}I(X_{d,1}) &= \frac{|w_1| + |Z_{1,32}|/2}{T_d} \\ &= \frac{|0.5| + |-0.2|/2}{2.9} = \frac{0.6}{2.9} \approx 0.2069\end{aligned}$$

$$\begin{aligned}I(X_{d,11}) &= \frac{|w_{11}| + |Z_{11,32}|/2}{T_d} \\ &= \frac{|2.0| + |0.2|/2}{2.9} = \frac{2.1}{2.9} \approx 0.7241\end{aligned}$$

$$\begin{aligned}I(X_{d,32}) &= \frac{|Z_{1,32}|/2 + |Z_{11,32}|/2}{T_d} \\ &= \frac{|-0.2|/2 + |0.2|/2}{2.9} = \frac{0.2}{2.9} \approx 0.0690\end{aligned}$$