Predicting Hint Usage in an Online Tutoring System

Jordan Argueta
North Carolina State University
Raleigh, NC
jiarguet@ncsu.edu

Jake Nowokunski
North Carolina State University
Raleigh, NC
jbnowoku@ncsu.edu

Prashan Sengo
North Carolina State University
Raleigh, NC
psengo@ncsu.edu

ABSTRACT

Many existing online tutoring systems allow students to request hints or other forms of help while working on various concepts and assessments. The ability to request assistance and receive feedback is critical to the learning process, however when used incorrectly, students hinder their progress rather than help facilitate it. Hints given at the incorrect time or not frequently enough when the student needs intervention can also lead to poor individual performances. In this paper, we analyze two datasets of student-tutor interactions to look for patterns in how students use hints and how hint usage relates to student performance. Being able to reliably predict when students will request a hint could enable online tutors to proactively provide help when needed. Our goal is to provide results that could help educators and tutor designers to craft hints that meet the circumstances under which they are used.

1. INTRODUCTION

Common practice in current tutoring systems is to implement hint tools that are primarily passive. This means the students request for hints rather than a tutor providing hints automatically when an individual becomes confused. This passive approach misses many benefits that a potential active tutor could have. Prior research indicates passive tutors are not usually used properly, and hints can cause students to spend more time on problems compared to students who do not utilize hints at all [4]. This phenomenon is seen when a dependency is formed between the student and the hint requests tool in order to avoid learning the course material. Behavior such as this reinforces incorrect learning techniques that can cause students to struggle to learn new concepts even outside of the online tutoring environment.

Prior research also demonstrates that students are not always at fault for hint abuse [1]. Hints given at the incorrect time or poorly integrated with the current step in the problem-solving process can cause students to consistently use the hint tool. Students may over use the hint system hoping to find the related information to their current knowledge gap. Knowing when a student needs help and where the student is at in the learning process is important for tuning the help functionality on all online tutoring environments. The issues around passive tutors are all areas that an active tutor aims to solve.

An active tutor would be similar to an in person tutor, providing assistance when the student shows signs of confusion or the need for further explanation before allowing the next attempt. An active tutor approach has the potential to lend students hints when they are under-utilizing hints or restricting students using hints when they are abusing them.

That being said, effective active tutors are difficult to create and implement correctly. In order to have a tutor that gives hints only when a student needs them, the tutor will need to have some representation of the students knowledge to see if they are stuck or not stuck. To properly create active tutors in intelligent systems, the primary objective is to explore how students are using hints in practice. Being able to accurately predict when students use hints will allow educators and tutor designers to create more responsive intelligent systems that provide help when a student needs it.

Prior efforts on hint prediction had a focus on tracking student knowledge and hint usage to determine when a student should seek help [2]. Each body of research used a different approach to knowledge tracing and a different method for hint implementation. Previous work lacked a significant body of research that focused on comparing and contrasting each implementation of knowledge tracing and hint prediction. Our goal is to compile a list of models found from previous works, use the models on the same datasets, and see how they compare in terms of resource performance, accuracy, and precision for multi-knowledge component multi-student datasets.

Through this paper we will start by doing a preliminary analysis on student hint usage within two tutoring systems. Our objective through this analysis is to try to find when hints are being used the most in a question. We will produce analytics through clustering for both datasets to visualize any groupings and dependencies that can be determined by student characteristics (time per question, attempts per question, etc) and their use of hints. Finally, we will use these characteristics to predict when a student will use a hint and evaluate all models' overall effectiveness compared to other approaches.

2. RELATED WORK

Educational Data Mining has been a growing area of interest in the research community, especially in online tools and resources. Discovering how students interact with online tutoring tools and other resources has been a key focus due to the potential to improve student interaction and involvement. In this section, we have collected and analyzed similar studies and approaches to help compare, contrast, and improve our own processes.

One of the key research questions that come with tutoring is how to offer on-demand help while influencing positive learning habits and outcomes. Previous studies have shown that learners who do not contain a related knowledge component before using a help-related resource are most likely to abuse a hint tutor system [2]. In most cases, students are either choosing to use resources not frequently enough or in some cases over utilize help without any positive growth on their behalf [9]. Research conducted around this topic aimed to identify what student related factors cause these types of interactions and how to avoid them [9]. Factors such as quality of resources, context, student control over the degree of help, and steps to the amount of cognitive load all contribute to how much the student will be willing to interact with various learning systems [9]. By utilizing this information, we actively found datasets that were similar in nature where the online tutor provided similar resources, degree of help, and amount of steps per question but also differed in subject matter content.

In prior works, datasets were created to compare students' progress in learning different environments; various groups were offered no hints while other groups were offered time-triggered clues and preemptive step-by-step tutorials [6]. The results showed that there was not a large difference to the assessment result which made it difficult to indicate whether or not hints provided any real benefit to the online learning service. One limitation identified by the study showed that there was not enough data to help isolate how students were interacting with the system. This helped in our data selection process to help alleviate some of the challenges encountered in previous studies. We purposely found multiple datasets similar in structure both with larger action sets to process.

More successful attempts of data analytics have been captured and successfully been turned into helpful/hint learning models like with the Cognitive Tutor for Algebra I [11] and the CALL system for Japanese language learning [13]. Both studies were able to collect a large dataset of student responses and capture how they were utilizing provided learning tools. While hints were not the main focus of either study, both were able to provide details on their preprocessing methods and steps that helped generate their research results. The main contributor toward our preprocessing methods was the work completed in the Cognitive Tutor Algebra I Program (Sales, Wilks, and Pane, 2016). In their selection of Imputing Missing Data, Sales et al, document the steps taken toward reducing the amount of inconsistent data and mitigate missing data values. Our data preprocessing approach adopted the same strategies to remove null data, identify model breaking entries, and balance the dataset.

2.1 Models Used In Hint Prediction

As stated in the introduction, our contribution is to compile a list of models found from previous works and provide comprehensive comparisons amongst the different models. We will compare how the models compare to themselves on different datasets and how the models perform against each other as well. The models selected were found from previous works that yielded positive results toward hint prediction in student-tutor learning environments.

Bayesian Knowledge Tracing or BKT is a common model used for student knowledge tracing and mastery estimation. BKT is similar to a Hidden Markov Model (MMM) as it contains observable nodes representing responses and hidden nodes that represent a student's knowledge at a given state or point in time [4]. Since its introduction from Corbett and Anderson it has become the control of modeling research, especially in the work done by Doung et al [7], and Chaudry et al [5]. Due to the common use of BKT, we also determined that it was imperative to include it into our data compilation as our baseline.

In the work presented by Doung et al [7], the authors used a Logistic Regression model to compare to BKT for student hint prediction. The goal of the paper was to use both BKT and Logistic Regression on a dataset from the ASSISTments online tutoring system. The result from the prediction accuracy showed that Logistic Regression outperformed BKT by 0.02 accuracy but under performed in MAE and RMSE scores by 0.02 [7]. After viewing the results, we wanted to see if this behavior would be replicated on datasets outside of the ASSISTments environment.

Random Forest and Decision Tree models were used in the work provided by Hawkins et al [8] and compared to the BKT model as well. The purpose of this paper was to find a model that would accurately predict if a student would answer the next question correctly. The results showed that Decision Trees scored 0.638 in accuracy while Random Forest scored 0.637 in accuracy. Hawkins et al found when compared to BKT, both models outperformed BKT as it scored 0.631 in accuracy [8]. Since both models yielded positive results when compared to BKT, it was important to apply these models to hint prediction and see how they compare to the rest of the models.

Support Vector Machine or SVM is the one model included in this body of work that did not have an example of previous work in the educational data mining space. However, SVM is commonly used in cost-sensitive environments such as Marketing and Security [12]. SVM performs well with mining data that is classified as highly unbalanced. The defining characteristics of SVM made it seem like a reasonable solution to hint prediction due to the frequency of unbalanced data found in previous studies.

How these models are implemented and defined will be further discussed in the Classifiers sections. The purpose of the list above was to summarize previous work that was done with these models. In summary, there are a multitude of resources that have covered prediction modeling for knowledge components predicting hint usage, however, to our knowledge there has not been any research information that compiles the results of these models on the same datasets.

3. DATASETS

The first dataset we will analyze is the "FrenchLanguage" dataset found on DataShop [10]. This dataset contains a log of student interactions with an online French language tutor. There are 102 unique students contained within the dataset and 436,493 logged transactions. As this dataset is very large, we should have enough data to train good supervised learning models.

The second dataset we will analyze is the "VectorComponentsTutor" dataset, also found on DataShop [10]. This dataset contains a log of student interactions with an online mathematics tutor. There are 20 unique students contained within the dataset and 2,783 logged transactions. This dataset is much smaller than the French dataset, and the results for this dataset will give us an idea of how our models will work in an environment where there is less data to train the models.

The French dataset keeps track of five different types of student interactions: starting the tutor, playing audio, playing video, making an answer attempt, and hint requests. The Vector dataset only has two types of student interactions: making an answer attempt and hint requests. For our experiments and analysis, answer attempts and hint requests will be the entry types we focus on.

Table 1: Overview for French Language Dataset and Vector Component Dataset

Summary:	**FrenchLanguage**	**VectorCompTutor**
Total Responses	436,493	2,783
Attempts	268,499	2,724
Hints	2,708	59
Unique KC	2,997	73

Table 1 compares the amount of total responses, the amount of attempts on problem set, hint usage, and unique knowledge components between the FrenchLanguage dataset and VectorComponentTutor dataset.

Each transaction in both datasets represents a student action within the online tutor. Each entry contains fields such as the anonymized ID of the student performing the action, duration to perform an action, the number of attempts that have been made, whether a student got a question correct or not, the number of hints requested, and more. These raw data points require some preprocessing in order to distinguish subproblems within multi-part problems so we can analyze the dataset and make predictions. We also need to remove some entries that are duplicate rows or contain incomplete information.

In both datasets, there is an extreme class imbalance between hint requests and normal student attempts. This will create a problem for our supervised learning models because simply predicting the majority class could result in a very high accuracy. To account for this we will attempt two imbalanced sampling techniques explored in the methods section of the paper.

4. METHODS

4.1 How Are Hints Used By Students?

We begin our analysis by examining the statistics of both hint usage and student performance in the datasets. We perform some basic analysis of the datasets, as well as analyzing some metrics computed from the datasets after preprocessing. The metrics that we calculate are explained in Section 5.1. Removing outliers from the datasets, such as students with a very low number of activities logged, will be completed before calculating these statistics. These statistics should give us an idea of average student behavior within the tutoring environments.

To perform this analysis, we first need to calculate each of the statistics we want to examine. Some of these statistics can be computed directly from the datasets, while others require more complex data processing. For example, to compute the number of attempts when using a hint versus not, we need to identify for each question if a hint has been used.

4.2 Clustering Students by Hint Usage and Performance

In order to cluster the students, we will need to compute a series of statistics for each individual student. These will be the same statistics we calculated for the entirety of the datasets, except calculated for each individual student. These statistics give us an idea of how often the student is using hints and how well they perform on the questions they answer. Again, students who have a very small number of entries will be excluded.

We use hierarchical clustering to cluster the students by a few of these statistics. Only a few of the statistics are used in order to avoid issues due to multicollinearity between statistics that are closely related. We will then analyze the statistics of students within each cluster to identify key differentiating characteristics for the clusters. We should be able to identify if the students in a cluster use hints more often than students in another cluster, or if students in a cluster use more attempts than students in another cluster.

4.3 Predicting Hint Usage

In the following section we will be looking at the steps we took to predict student hint usage within our dataset. Primarily, we will be looking at the general overview, inputs, classifiers, sampling techniques and evaluation design decisions we used for predicting student hint usage.

4.3.1 General Overview

For the predictions of the hints there are several steps and transformations that were made. First, for the preprocessing and tuning stage we took the data and computed for each entry several values such as what subproblem the student is on, the number of hints used so far for the subproblem, and other features we will discuss later on in Section 4.3.2. Then we did some feature transformation, normalizing any continuous values and one-hot encoding categorical values. Finally, we applied grid search to tune the hyperparameters for our classifiers, checking several combinations of hyperparameters for each classifier and selecting the hyperparameters that result in the best value for the performance metric.

For the classification stage we performed 5-fold cross validation for the training and testing of the classifiers and generated the predicted hint usage values for each row in the datasets. Since the datasets had highly-imbalanced classes, we also applied some sampling techniques to the training data such as SMOTE and Under-sampling to even out the distribution of the classes in the training data. We compare the prediction results using the sampling techniques and without using the sampling techniques to evaluate their effectiveness.

Finally, for the evaluation stage we took the prediction results of whether a student used a hint or not for the given input in a classifier and computed several evaluation metrics including accuracy, recall, precision, and more for the minority class. We also drew a ROC curve to visually compare the performance of the models between each other.

4.3.2 Inputs

For the inputs of our prediction we tried to use past behavior of the student/problem to predict future hint usage behavior. The particular inputs that were used along with their explanations are listed below:

- **Subproblem:** This variable gives information on what subproblem the student is on for the problem they are doing. This helps us identify the specific part of the problem that the student is working on.
- **Duration (sec):** This variable gives information on how long the student takes for the step they are on.
- **Attempt Number:** This variable gives information on the attempt number the student is on for the current subproblem.
- **Hints so Far:** This variable gives information on how many hints the student has used so far for the subproblem they are on.
- **Is Last Attempt:** This variable gives information on whether the student is on their last possible attempt for the subproblem.
- **KC(Unique - step):** This variable gives information on the particular knowledge component associated with the current subproblem.
- **Outcome**: This variable tells whether the student's attempt was incorrect or correct. It will only be used for the baseline model.

4.3.3 Classifiers

For our prediction we have tested five classifiers, with four of the classifiers being in our experimental group and one being the baseline that we will compare our results to. For the prediction of hint usage we compare the different fundamental approaches between the baseline and experimental groups to determine which is better at predicting hint usage. The classifiers and their respective descriptions are listed below:

Baseline Model:
The baseline classifier predicts the students hint usage based on how well they know the learning material. In other words, we assume if a student has a high probability they know a particular learning material then we assume they will not request a hint as they already know the material. This was chosen as the baseline as it naively assumes their is a deep correlation between student knowledge and hint usage along with being a model that has been used in several related works to provide hints when a student needs them [5] Listed below is more information on the particular classifier used for the baseline.

- **Bayesian Knowledge Tracing:** BKT models student knowledge on a particular knowledge component using a Hidden Markov model [4]. It assumes that knowledge is broken down into atomic skills that a student either has or has not learned. Using the Knowledge component, Problem, and Outcome of a student/problem pair as inputs it can give a probability on how likely the student is to know the given Knowledge component for the problem and update this internal model of the students knowledge based on the actual outcome of the problem.

Experimental Model:
The experimental group attempts to model a students hint usage through the students prior hint usage and behavior. This approach allows for the student's hint use (or lack thereof) to be better modeled through the use of prior hint taking behaviors. We selected some general binary classifiers to model the students behaviors with specific consideration put into choosing models that are good for imbalanced datasets and would run relatively fast.

- **Logistic Regression:** Logistic Regression gives the probability of the output using a mathematical combination of the inputs. This classifier is particularly useful in modeling binary values as any probability above 0.5 can be classified as one class while anything below is classified as the other class.

  This classifier was selected for its ease of handling large datasets along with being less likely to overfit which is important as the imbalance of actions where hints were and weren't used could result in overfitting towards the majority class (when hints are not used).
- **Support Vector Machines:** Support Vector Machines operate by plotting the inputs into a graphical space and drawing a hyperplane that best minimizes the errors in classification. All the points on one side of the hyperplane are classified one way while all the points on another side of the hyperplane gets classified another way, allowing for binary classification.

  We chose SVM for its ability to handle sparse data which occurs in our dataset from one-hot encoding the knowledge components.
- **Decision Tree:** Decision Trees is a classifier that uses a tree structure where splits are made in the tree according to different criteria composed of values of the input variables. When the input values are fed into the tree the values follow the branches that satisfy the criteria for the nodes until the leaf node is reached which outputs the classification value for the given inputs.

Decision Trees were chosen mostly for the fact that feature selection happens automatically which is useful in our dataset which has thousands of features from one hot encoding the knowledge components. This allows for a reduction in the features that need to be looked at allowing only the most important features to determine the outcome of a particular input.

- **Random Forest:** Random Forest is an ensemble classifier that uses many decision trees that aggregate their results to produce the classification value for a given set of input values.

  Random Forest has the same benefits as Decision Trees with the extra benefit of reducing overfitting and allowing for more complex decision boundaries.

4.3.4 Sampling Techniques

Since our datasets are highly imbalanced (97% of data points don't use hints while 3% use hints) we have a couple techniques that we have selected to train our classifiers in a more balanced way. The two methods we will be evaluating are SMOTE and Under-sampling. By using these techniques we can prevent our classifiers from being overfitted to the majority class and get better predictions for the minority class.

- **Under-sampling:** Under-sampling is a method of sampling where you sample all the minority class in a training set and sample the same amount of majority class in the training set and use the resulting sampled set as the training data. For example if the training data had 300 true cases and 1 million false cases then the resulting sampled training data has 300 true cases and 300 false cases. We use under-sampling for the French dataset as it contains a large amount of rows with a majority of rows having the target value of no hints used. Using under-sampling on this data reduces the computation time and can make the models more generalized and less tuned to the majority class.

- **SMOTE:** SMOTE is a method of sampling where new values from the minority class are synthesized based on preexisting values in the minority class. This creates a resulting sample that has the same amount of minority and majority classes in the sample where the synthesized minority class has values close to the original minority class. We use SMOTE in the Vector dataset as SMOTE will typically double the amount of samples you have so it is a good approach for data sets with a small amount of samples like in the vector component dataset.

5. RESULTS

5.1 How Are Hints Used By Students?

To evaluate how hints are used within each dataset, we computed several metrics relating to how students perform on each subproblem and the circumstances under which they decide to use hints.

Prior to computing these statistics, we removed students from each dataset that had a total number of entries less than a threshold. This threshold differed for each dataset. In the French dataset, students with less than 50 entries in the dataset were removed. 20 students were below this threshold, and our analysis focused on the remaining 82 students. In the Vector dataset, students with less than 20 entries in the dataset were removed. This threshold was lower in comparison to the French dataset due to the lower average amount of entries for each student. 2 students were below this threshold, and our analysis focused on the remaining 18 students. The Vector dataset also contained duplicate information for every row, and these duplicate entries were removed.

The results for each dataset are shown in Table 2.

Table 2: Computed statistics of each dataset for average student attempts, average duration per subproblem, and average hints used per subproblem.

	French	**Vector**
Avg Attempts	1.395	1.049
Avg Attempts (hint used)	7.271	2.375
Avg Attempts (no hint)	1.364	1.045
Avg Duration	15.187s	6.745s
Avg Duration (hint used)	114.778s	42.875s
Avg Duration (no hint)	14.690s	6.616s
Hints per Subproblem	0.013	0.014

"Avg Attempts" represents the average number of attempts that students make on subproblems. If the average was 1.000, that would mean students always take one attempt to complete a subproblem, and every attempt was a correct answer. If students are making incorrect attempts, they will have additional attempts to complete the problem, resulting in a number of attempts greater than 1. From the results in Table 2, we can see that in the French dataset, students take on average between one and two attempts to complete a subproblem. In contrast, the average number of attempts in the Vector dataset is much closer to one. This suggests that subproblems in the Vector dataset are often completed in only one attempt, and students are often getting problems correct on the first try.

Students using a hint or not using a hint has a similar effect on the average number of attempts in both datasets. When students do not use a hint on a subproblem, the average number of attempts is slightly lower than the average number of attempts for the entire dataset. This makes sense due to the majority of subproblems in the datasets not having a hint be used. When a student does use a hint, the average number of attempts is much greater than the average number of attempts for the entire dataset in both datasets. This suggests that students are often using hints in situations where they are struggling with a problem and have previously made some incorrect attempts.

"Avg Duration" represents the average duration that students take to complete subproblems. On average, students take about 15 seconds to complete a subproblem in the French dataset, and students take about 7 seconds to complete a subproblem in the Vector dataset. This difference makes sense as students in the French dataset are completing language-based problems in a language that is likely not native to them, so it will take more time for them to understand the problem and come up with a solution.

Students using a hint or not using a hint has a similar effect on the average duration as it does on the average number of attempts. Because students are often using more attempts when they opt to use a hint, it makes sense that the average time it takes for them to complete a subproblem where they use a hint will be much longer than the average duration for the entire dataset.

”Hints per Subproblem” represents the average number of hints used by students on each subproblem. If this average were 0, it would mean that no hints are used on any subproblems in the dataset. We can see from Table 2 that a similar number of hints are used per subproblem in both datasets, and that this proportion is very low.

5.2 Clustering Students By Hint Usage and Performance

The goal of using clustering with these datasets is to identify groups of students with similar behaviors while using the online tutors. Once we have these groups, we can analyze the statistics of students within them to see if these clusters can be used to give meaningful insights about how different groups of students use hints.

To begin our analysis for this section, we computed the same statistics as shown in Section 5.1 on a per-student basis. These statistics give us an idea of how many attempts each student takes on average, how long they take to answer subproblems, and how often they use hints.

If we were to use all of these statistics to create the clusters, there could be issues in the results due to multicollinearity between several of the statistics. For example, ”Avg Duration” and ”Avg Duration (no hint used)” will be highly correlated due to subproblems where no hint is used being the majority of cases, resulting in very similar values between these two statistics. For students who use no hints, these values will be exactly the same. To address this potential issue, only three of the computed statistics are used to compute the clustering: ”Avg Attempts,” ”Avg Duration,” and ”Hints per Subproblem.” However, after the clustering is completed, we can still use all of the computed statistics in order to analyze the characteristics of students within each cluster.

Prior to clustering, students who used no hints are assigned a value of 0 for ”Hints per Subproblem.” Also, all statistics used are min-max scaled to be within the range [0, 1]. This prevents the magnitude of the statistic values from inadvertently weighting the influence of each statistic.

Originally, we planned to use K-Means to cluster the students. However, upon attempting to find the best-fit value for the number of clusters to create for the French dataset, we found no clear number of groups that created clearly-distinguishable clusters. Upon visual inspection of the data, we found that the majority of students in this dataset belonged to one large cluster, with only a few notable outliers. As a result, we decided to use hierarchical clustering with single linkage to better fit the dataset.

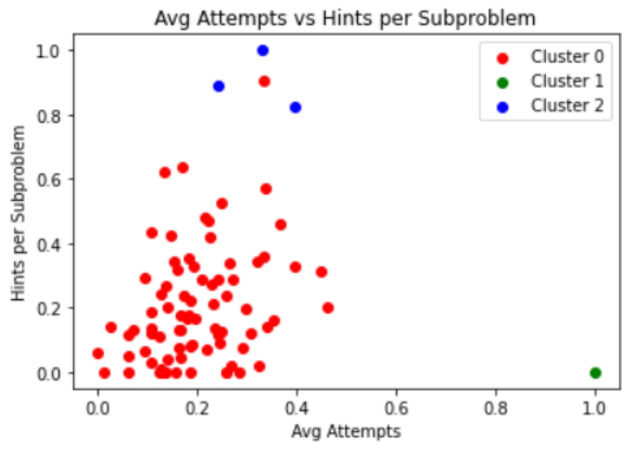

Figure 1: Clusters of students in FrenchLanguage dataset, plotted by average number of attempts and average hints used per subproblem, scaled to the range [0, 1].

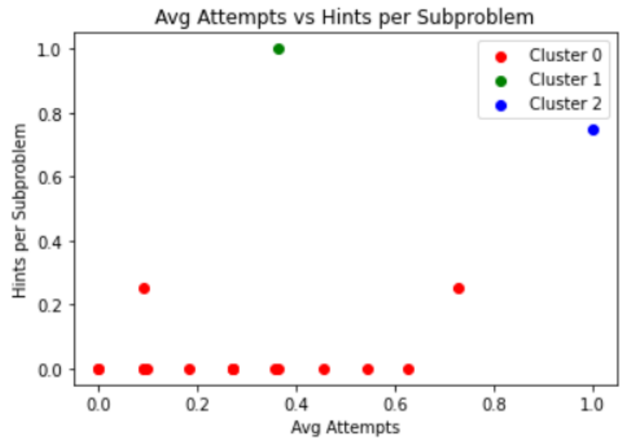

Figure 2: Clusters of students in VectorComponentsTutor dataset, plotted by average number of attempts and average hints used per subproblem, scaled to the range [0, 1].

The results of our clustering are discussed in the following subsections.

5.2.1 French Dataset

As discussed previously, we decided to use hierarchical clustering to create the most clearly-distinguishable clusters possible for the French dataset. The computed statistics ”Avg Attempts,” ”Avg Duration,” and ”Hints per Subproblem” for each student were used to create the clusters. Upon visual inspection of the data, we decided that three clusters would be the most appropriate number of clusters to create due to there being only two clear ”clusters” outside of the large cluster containing the majority of students in the dataset.

The results of the clustering for the French dataset are shown in Figure 1. This figure shows each student plotted by their ”Avg Attempts” and ”Hints per Subproblem,” scaled to the range [0, 1], as plotting these two variables can visually capture much of the difference between the three clusters. As shown in the figure, there are three distinct clusters. The largest cluster, Cluster 0, contains the majority of students in the dataset. Cluster 1 contains only a single student with a high average number of attempts and no hints used. Cluster 2 contains three students with a high number of hints used per subproblem.

Table 3: Computed statistics within each cluster for the FrenchLanguage dataset.

Cluster	**0**	**1**	**2**
Avg Attempts	1.374	2.585	1.560
Avg Attempts (hint used)	8.519	N/A	5.644
Avg Attempts (no hint)	1.344	2.585	1.472
Avg Duration	14.944s	3.707s	25.355s
Avg Duration (hint used)	115.133s	N/A	106.730s
Avg Duration (no hint)	14.486s	3.707s	23.659s
Hints per Subproblem	0.012	0.000	0.052

Table 3 shows the average of the computed statistics for the students within each computed cluster.

As Cluster 0 contains the majority of the dataset, the student values within this cluster are very similar to the statistics for the entire dataset. These students typically used very few hints and had a low average number of attempts. In scenarios where they do decide to use hints, they often make several attempts at the subproblem. This suggests that the majority of students in this dataset use hints to help them with subproblems where they may be struggling to come up with the correct answer.

Cluster 1 contains only a single student. This student used no hints, had a relatively high average number of attempts, and had a relatively low average duration per subproblem. Upon further inspection of this student's data, the student completed only one section from one unit of the course and made 106 answer attempts, of which only 21 were correct. This student appears to have either been struggling to complete the questions or was carelessly making several quick, incorrect attempts. In either case, using hints or having hints suggested may have improved this student's performance.

Cluster 2 contains the remaining three students. These students had a higher average number of attempts, higher average duration, and considerably higher average number of hints used per subproblem when compared to the majority of the dataset. Notably, their average number of attempts when a hint is used is also considerably lower than the majority of the dataset as well. These statistics suggest that these students may have been much more willing to use hints in situations where they are unsure of the answer.

5.2.2 Vector Dataset

To compare the use of clustering in the French dataset to the Vector dataset, we again use hierarchical clustering with single-linkage and three clusters. The results of the clustering for the French dataset are shown in Figure 2. Cluster 0 contains the majority of students in the dataset, while Clusters 1 and 2 contain only a single student in each cluster.

Table 4 shows the average of the computed statistics for the students within each computed cluster.

Table 4: Computed statistics within each cluster for the VectorComponentsTutor dataset.

Cluster	**0**	**1**	**2**
Avg Attempts	1.042	1.056	1.153
Avg Attempts (hint used)	1.250	3.000	4.000
Avg Attempts (no hint)	1.042	1.028	1.113
Avg Duration	6.617s	7.181s	8.347s
Avg Duration (hint used)	17.5s	62.0s	74.5s
Avg Duration (no hint)	6.579s	6.408s	7.415s
Hints per Subproblem	0.003	0.111	0.083

Similar to the French dataset, Cluster 0 contains the majority of students in the Vector dataset. The average values for students in this cluster are very similar to the average values for the entire dataset. The majority of students appear to use very few hints.

Cluster 1 contains a single student who used the most hints per subproblem on average. Cluster 2 contains a single student who used the second-most hints per subproblem on average and the most attempts per subproblem on average. Upon further inspection of the data, both of these students used several hints for a single subproblem and used no hints for the remaining subproblems. In the statistics for when hints are not used, their values are very similar to the values for the majority of the dataset. However, their average attempts and duration when a hint is used are much higher than for the majority of the dataset. This suggests that their statistics were skewed by specific instances where they struggled with a problem and requested hints. This effect is exacerbated by the fact that each student has very few entries in this dataset relative to students in the French dataset.

5.2.3 General Insights

Overall, clustering students using these statistics displays limited effectiveness at identifying clusters of students who tend to either abuse hints or are not using hints and possibly should be provided them.

In both datasets, the majority of students fall into a single large cluster of students who use hints sparingly and perform relatively similar in terms of how many attempts they require to complete each subproblem.

The remaining clusters consist of only a few students who are outliers relative to their peers, either in terms of hint usage or the average number of attempts they require to complete subproblems. In the French dataset, these clusters of outliers could potentially indicate students who use hints more liberally or students who are struggling and may require more assistance. However, in the Vector dataset which has less data per student, the outliers are students who had some anomalous behavior that skewed their statistics. In either case, these outliers could be identified using methods other than clustering.

5.3 Predicting Hint Usage

In this section we will be looking at whether predicting using a student's behavior is better than predicting using a student's knowledge when it comes to hint usage prediction. To achieve this we use a BKT to model the students knowledge and predict whether they will use a hint and we use the inputs described in Section 4.3.2 to model the students behaviors and predict using the four classifiers mentioned earlier. We examine how the two approaches compare to one another along with hypothesizing why the results are the way they are. Also we look at whether the performance for the behavior driven approach can be improved through the use of imbalanced sampling techniques for the classifiers.

5.3.1 BKT Vs Behavior Classifiers

Using both the French Language and Vector Component datasets, we have taken each classifier mentioned earlier and predicted whether a student used or didn't use a hint for each row in the dataset. From these predictions we computed the Accuracy, Precision, Recall and F1-score of the classifier's prediction and outputted the values into a table for comparison. We mainly focus on maximizing precision, recall and F1-score as it tells us how well the classifiers perform at predicting the minority class (case where student used a hint).

Table 5: Performance values for FrenchLanguage and VectorComponentTutor datasets.

Classifier	Accuracy	Precision	Recall	F1-Score
French Language (Big dataset)				
BKT	0.2421	0.0041	0.3116	0.0081
Random Forest	0.9915	0.5907	0.4713	0.5243
Logistic Regression	0.9903	0.6027	0.0526	0.0968
SVM	0.9904	0.7015	0.0562	0.1041
Decision Tree	0.9895	0.4736	0.4821	0.4778
Vector Component (Small Dataset)				
BKT	0.2453	0.002	0.1111	0.0038
Random Forest	0.9927	0.7857	0.6111	0.6875
Logistic Regression	0.9898	0.6	0.6667	0.6316
SVM	0.9898	0.5833	0.7778	0.6667
Decision Tree	0.9905	0.6667	0.5556	0.6061

Taking a look at the values in Table 5, we see that BKT values did not outperform against any of the classifiers used in the behavior approach for either datasets. We saw that out of all the cases where a student used or didn't use a hint only 25 percent were correctly predicted (accuracy) in both datasets compared to the classifiers in the behavior approach that correctly predicted 98 percent of the cases. We also see that out of the amount of cases where a hint was actually used, only about 11 to 31 percent of the predictions were predicted as hint used (recall) using BKT. This is compared to some of the classifiers like Random Forest and Decision Tree which had values from 40 to 80 percent for recall. We see this pattern repeat for other metrics; for example, out of all the cases where a hint was predicted to be used only 0.2 to 0.4 percent were actually deemed to be used (precision) for BKT. This was not seen in the behavior classifiers which ranged from 47 to 79 percent for precision. The poor performance of BKT may indicate that a model using student knowledge to predict hint use may be worse than using one that predicts off of prior behavior.

Some reasons that we see such a poor performance in BKT compared to the other classifiers could be from the nature of the classifier. In BKT we are assuming when a student doesn't have high knowledge in the knowledge component for a given question, the student will ask for a hint because they need a hint at that moment and we make the naive assumption that the student will want a hint every time they need a hint. This results in the classifier labeling nearly all the early rows in the dataset as the student requesting a hint because the students' latent knowledge of the KC's are low at that time, indicating that the student needs a hint and thus that they want a hint. The other classifiers do not have such a nearly profound cold start problem as they are only predicting off of the inputs given to them and don't need to rely on prior data.

5.3.2 Best Classifier in Behavior Classification

In this section we will be looking at which classifiers from the 4 classifiers we have tried in the behavior approach perform the best on both datasets. We will be using the precision recall curve, which shows the trade-offs between precision and recall for a variety of thresholds and can help us find the best classifier that minimizes the true negatives and false positives of students who use hints.

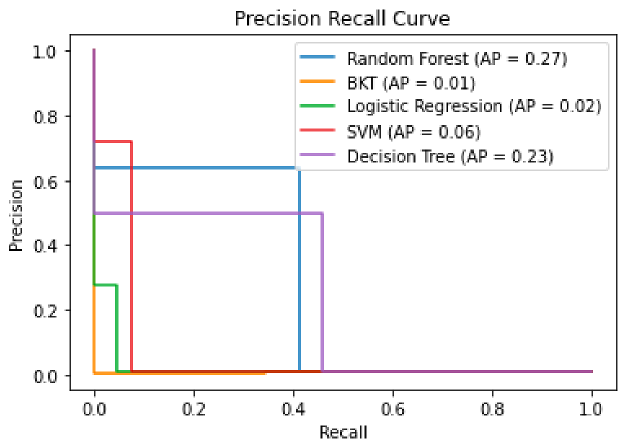

Figure 3: Precision Recall curve of classifiers in FrenchLanguage dataset.

Looking at Figure 3, we see that for the French dataset the Random Forest and Decision Trees classifiers seemed to outperform the Logistic Regression and SVM, as seen by the fact that they are closer to the ideal scenario of having precision and recall equal to 1. We have also calculated the average precision to represent the overall performance of the classifier and as we can see, the Decision Tree and Random Forest outperform the other classifiers by more than 20 percent. When we look at Figure 4 on the Vector component dataset we see that Random Forest and Decision Trees are still the best however SVM and Logistic Regression which used to be the worst classifiers are now comparable in performance to the others. The average precision values for the vector component dataset also reflects the mentioned pattern.

Having looked at the results, it is fairly clear that Random Forest is the best classifier as it has maximized the precision and recall in the precision recall curve. One possible explanation for this result is that the ensemble nature of Random

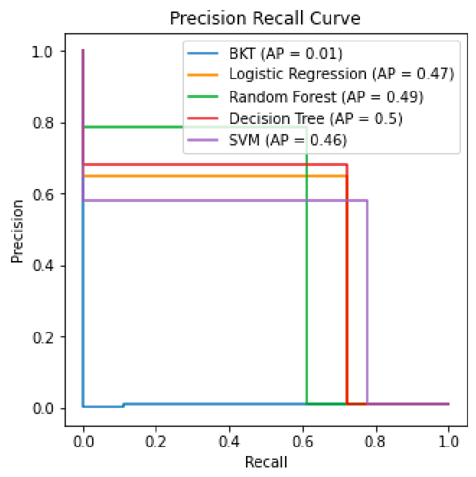

Figure 4: Precision Recall curve of classifiers in Vector Component dataset

Forest makes it better at capturing and representing information from the minority class which other classifiers could easily overlook due to the small number of cases where hints were used. We also see that the variation in performance of the classifiers may be related to the size of the dataset. In particular, it seems classifiers whose outputs are a mathematical combination of its inputs like Logistic Regression and SVM seemed to perform worse for the larger dataset (Figure 3) but perform better in the small dataset (Figure 4).

5.3.3 *Sampling Techniques*

Table 6: Performance values for FrenchLanguage and VectorComponentTutor datasets after sampling techniques.

Classifier	Accuracy	Precision	Recall	F1-Score
French Language Classifiers after Under-sampling				
Random Forest	0.9472	0.1555	0.9749	0.2682
Logistic Regression	0.8908	0.0805	0.9605	0.1486
SVM	0.9389	0.1342	0.9462	0.2351
Decision Tree	0.9441	0.1448	0.9438	0.2511
Vector Component Classifiers after SMOTE				
Random Forest	0.9934	0.7647	0.7222	0.7429
Logistic Regression	0.9818	0.4	0.7778	0.5283
SVM	0.9731	0.3061	0.8333	0.4478
Decision Tree	0.9913	0.65	0.7222	0.6842

In the Table 6 we see the effects of applying Under-sampling on the French dataset and SMOTE on the Vector dataset. In the table a red value means that the value is at least one percent less than the original value in Table 5, while the green values mean the value in the table is at least one percent greater than the original value in Table 5. The black values mean there was no change or less than a 1 percent change in value. From the table we can see a couple of patterns. Starting with the Under-sampling section of the table, we see that applying Under-sampling to the French dataset resulted in generally a lower accuracy and precision while it generally increased recall and F1-score. When we look at implementing SMOTE on the Vector dataset, we see that the accuracy stayed relatively the same while precision dropped. It was also seen that recall typically went up and F1-score was typically split on whether it went up or down.

To understand why we see the results we do for the Under-sampling section (Table 6) it is important to understand what Under-sampling does. Under-sampling works by taking the training data and training using all the values in the minority class and the same amount of values in the majority class, essentially meaning that less values in the majority class are being trained with in the classifier than originally trained with in the original dataset.

Training using less rows of the majority class means that the classifier is less precise in making classifications to the majority class resulting in more predictions to the minority class. This is because some of the rows the classifier would have predicted as the majority class would get predicted as the minority class as the model is more generalized after Under-sampling. The effect of more predictions to the minority class would be a lower accuracy. Because of the imbalanced nature of the dataset, you would be more likely to be correct if you predict the majority class than the minority but since we are predicting more minority class values we can expect more misclassified points. We also see that the more generalized classifier will result in a higher true positive rate for predicting the minority class (more predictions increases chance of predicting correct) along with a higher false positive rate for the minority class (more predictions also means more chance of incorrect predictions). The effect of increasing the true positive and false positive rate is an increase in recall and precision based on their formulas.

The same reasoning can be seen in the SMOTE section except that instead of reducing the amount of majority class training samples we are increasing the minority class training samples. This essentially causes the classifier to be more precise towards predicting the minority class, causing the true positive value of the minority class to increase thus increasing the recall. Since SMOTE synthetically creates data points off of the minority class from actual minority class data we could expect some of these points to fall outside the actual distribution of the minority class. This can be where some of the misclassification of predicting the minority class can occur, resulting in an increase in the false positive rate thus resulting in the precision and accuracy to decrease based on their formulas.

5.3.4 *General Insights*

Based on our results, there are a handful of key observations that we have found from our analysis:

- Modeling using student behaviors seems to be better than modeling using student knowledge as modeling on student knowledge seems to have a cold start problem.
- Out of the classifiers we have tried, Random Forest seems to be the best classifier for predicting hint usage. This is likely because the ensemble nature allows for more representation of the minority class.
- Applying imbalanced sampling techniques didn't result in an overall improvement of the classifier's performance; rather, it traded off between recall and precision.

6. CONCLUSIONS AND LIMITATIONS

In this paper, we have provided a detailed analysis of two datasets, clustered students by usage and performance, and compared the results of hint prediction of several classification models. As the section prior had highlighted, both Random Forest and Decision Tree outperformed the rest of the classifiers used in both precision and recall scores. This supports the results found in Hawkins et al [8], where both Random Forest and Decision Tree models outperformed BKT when tested against the same datasets. Logistic Regression recall and F1-scores were the most inconsistent between the two datasets, varying from 0.0526 to 0.6667 for recall and 0.0968 to 0.6875 for F1-score. It seems in these categories, Logistic Regression performed better when the dataset was smaller. SVM showed similar results to Logistic regression with highly varying recall and F1-scores, seeing higher performance overall on the smaller dataset. However, since SVM is a model that excels with imbalanced datasets it was expected the model perform better on the "VectorComponentTutor" dataset [12].

When attempting to cluster students based on their hint usage and performance in the datasets, our results proved to have limited effectiveness. The majority of students fall into a single large cluster, while the remaining clusters only contain a few outliers. In the larger French dataset, these outliers may be indicative of students who either may need more hints or are using hints too liberally. For the smaller Vector dataset, these outliers have only a single instance of using several hints skewing their statistics and are less indicative of habitual hint abuse. Using the statistics we calculate, other methods designed for outlier detection in student behaviors may be better suited to this problem.

The results produced for classification can provide further insight into which of the common classifiers used in prior research would produce the most consistent performance results in the future. While we are confident in our findings and process, there are areas in our research that we would improve in the future. One of the limitations that was identified through our prediction modeling process was that the Vector dataset was insufficiently sized. With only 2,783 responses total, and even less after preprocessing, the Vector dataset was not able to confidently produce a model as accurate compared to the French dataset. As with prior works, there is a concern that small datasets can result in poor approximation and prediction results as seen with C. Cody and T. Barnes research on their Deep Though hint prediction tool [6]. We used the results from the Vector dataset as a comparison to the classification models created from the French dataset, however, the comparison is not as dependable as we were hoping for. This can be seen in the drastically differing recall and F1-scores between the two groups of classifiers.

While the Vector dataset was relatively small, it was not the only dataset that had difficulties. Through our attempts at using BKT to produce prediction results, it became apparent that both datasets contained an extremely low number of responses containing hint usage. Since there was not enough adequate data to properly train BKT, all of the prediction results for BKT under performed when compared to BKT models from previous work. When compared to other classifiers from both datasets, BKT performed the lowest with an accuracy score of 0.2421 for the "FrenchLanguage" dataset and 0.2453 for the "VectorComponetsTutor" dataset. The next lowest accuracy score was Decision Tree and Logistic Regression for each respective dataset. Compared to the results produced by Duong et al in their comparison of a Linear Regression model to BKT, the average difference in accuracy was 0.003 [7].

We also saw the effects of a small subset of hint responses when our attempts to improve model performance with sampling produced results with lower scores than without sampling, as seen in 6. However, we were not able to apply the same sampling techniques to both datasets for an accurate comparison between the two results for the classifier groups. For the SMOTE sampling technique, the French dataset was too large to process all the data points causing resource exhaustion error on our run server. This was the main reason behind the decision to use Under-sampling for the French dataset. When we applied Under-sampling to the Vector dataset there were too few data points to train with, and for that reason we decided to use SMOTE for the Vector dataset.

In our original plans we had another classifier that we wanted to utilize in our research. In Chaudhry et al work, they used a Neural Network as their baseline classifier to test out their Colearn model results [5]. We had found the results promising and wanted to see how the Neural Network model would compare to BKT, Random Forest, etc. Due to resource constraints on our run server we were not able to complete a run of a Neural Network for the French dataset due to its size. As a result of this constraint we decided to omit results for the Neural Network model from this report.

7. FUTURE WORK

This work is just a small part of the overall goal toward improving hint prediction for online tutors. One of the objectives of this research is to provide a comparison across commonly used classifiers to have a clear representation of how they compare to one another across various datasets. In the future, we would like to work with numerous varying datasets related to online tutors in order to provide consistency and repeatability to our results and process. Our datasets provided us with numerous challenges that could be mitigated in the future with larger datasets or different models. Moving forward, future work also includes addressing resource constraints in order to expand on the amount of classifiers we can explore and utilize. In order to have a wider range of diversity in our results and analysis, future work must include improved results for BKT that fit closer to prior work and results for Neural Network models. An additional area we did not have the opportunity to explore further was the comparison of various sampling techniques and their impact on the performance of different classifiers. While there are many other areas to explore and improve upon, it is our desire that this research be used in future work with the objective to improve hint prediction performance where the result presented in this paper are the baseline for comparison.

8. REFERENCES

[1] V. Aleven, E. Stahl, S. Schworm, F. Fischer, and R. Wallace. Help seeking and help design in interactive learning environments. *Review of Educational Research*, 2003.

[2] V. Aleven, E. Stahl, S. Schworm, F. Fischer, and R. Wallace. Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artifical Intelligence in Education*, 2006.

[3] A. Badrinath, F. Wang, and Z. Pardos. pybkt: An accessible python library of bayesian knowledge tracing models. *Educational Data Mining Org*, 2021.

[4] R. S. Baker, A. T. Corbett, and V. Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. *International conference on intelligent tutoring systems*, 2008.

[5] R. Chaudhry, H. Singh, P. Dogga, and S. K. Saini. Modeling hint-taking behavior and knowledge state of students with multi-task learning. *11th International Conference on Educational Data Mining*, 2018.

[6] C. Cody and T. Barnes. Analyzing the associations of hint type on student behavior and performance. *International Educational Data Mining Society*, 2018.

[7] H. D. Duong, L. Zhu, Y. Wang, and N. Heffernan. A prediction model uses the sequence of attempts and hints to better predict knowledge: Better to attempt the problem first, rather than ask for a hint. *Computer Science Department at WPI*, 2014.

[8] W. Hawkins, N. Heffernan, Y. Wang, and R. S. Baker. Extending the assistance model: Analyzing the use of assistance over time. *Educational Data Mining Org*, 2013.

[9] S. Karumbaiah, J. Ocumpaugh, and R. Baker. The influence of school demographics on the relationship between students' help-seeking behavior and performance and motivational measures. *International Educational Data Mining Society*, 2019.

[10] K. Koedinger, R. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the edm community: The pslc datashop. In *Handbook of Educational Data Mining*, Boca Raton, FL, 2010.

[11] A. Sales, A. Wilks, and J. Pane. Student usage predicts treatment effect heterogeneity in the cognitive tutor algebra i program. *International Educational Data Mining Society*, 2016.

[12] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser. Svms modeling for highly imbalanced classification. *IEEE Transactions On Systems*, 2008.

[13] C. Waple, H. Wang, T. Kawahara, Y. Tsubota, and M. Dantsuji. Evaluating and optimizing japanese tutor system featuring dynamic question generation and interactive guidance. *School of Informatics, Kyoto University*, 2007.