Adaptive Teaching: Learning to Teach

by

Aazim Lakhani
Bachelor of Computer Engineering, University of Mumbai, Mumbai, 2009

A Project Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Adaptive Teaching: Learning to Teach

by

Aazim Lakhani

Bachelor of Computer Engineering, University of Mumbai, Mumbai, 2009

Supervisory Committee

_____

Dr. Nishant Mehta, Supervisor

(Department of Computer Science)

_____

Dr. George Tzanetakis, Departmental Member

(Department of Computer Science)

ABSTRACT

Traditional approaches to teaching were not designed to address individual student's needs. We propose a new way of teaching, one that personalizes the learning path for each student. We frame this use case as a contextual multi-armed bandit (CMAB) problem a sequential decision-making setting in which the agent must pull an arm based on context to maximize rewards. We customize a contextual bandit algorithm for adaptive teaching to present the best way to teach a topic based on contextual information about the student and the topic the student is trying to learn. To streamline learning, we add an additional feature which allows our algorithm to skip a topic that a student is unlikely to learn. We evaluate our algorithm over a synthesized unbiased heterogeneous dataset to show that our baseline learning algorithm can maximize rewards to achieve results similar to an omniscient policy.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

Dr. Nishant Mehta for his dedication, commitment and insights, without which I would not have been able to overcome the gaps I could not see.

Almighty One for giving me the courage, belief and strength to pursue my ideas.

Mom for her blessings and prayers.

DEDICATION

I dedicate this project to my family, to whom I owe both the joy and pain of growing up.

Chapter 1

Introduction

The development of a personalized learning system began with the creation of an intelligent tutoring system (ITS) [7, 20, 39, 41]. ITSs teach students in different ways based on the pedagogical rules defined for each student. Each student is categorized into groups and taught based on rules defined by the instructor. Such systems would use pre-existing knowledge through assessments, recognize different learning styles, and the track students progress to behave intelligently.

Each ITS typically comprises of the following components which interact with each other to teach a student. The content component maps concepts which are taught with the prerequisites and dependencies required to understand the concepts. The student model categorizes students into groups to track and categorize each student into groups. The pedagogy component defines instructions delivered to a student. This component analyses responses to assist students by providing content based on the pedagogical rules defined within it. The pedagogical component presents content based on the student group.

There are a few reasons why ITSs are not ubiquitous in education. They are primarily rules-based and need domain experts to manually specify every possibility that a system might face so it could present appropriate learning actions. They are non-adaptive, that is, they provide the same response to a problem independent of a student's previous interactions with the system. Examples are such system include German Tutor [18] and SQL-Tutor [28]. ITSs assume the student's behavior to be stationary, which is not true in the real world. To manage pedagogical rules, deliver content and categorize students is labor-intensive and time-consuming.

In recent years, machine learning has shown that it has the potential to personalize learning and scale for several courses and students. Machine learning based systems

[38] use data to personalize learning actions for each student without the need to explicitly specify learning actions for each student. Example of actions could be reading a chapter from a book or article, listening to a podcast, watching a video, or interacting with the system by answering quizzes. These systems are continuously learning from the data generated through students interactions with the system. Thus it has the potential to eliminate the challenges one would face with a traditional ITS [22].

The goal of this project is to design a learning algorithm, which could adapt based on student's feedback to help them learn effectively.

1.1 Use Case

There is no universal best way to explain a topic. The best way is subjective to every student. Unless we explore different ways to teach a topic, we cannot find the best ways to teach different students. Every student is unique, so we need to use different teaching methods to find a way that is conducive for a student. Once, we have found it we can use this knowledge to teach other similar students effectively. **This is the exploration-exploitation dilemma where there is a trade-off between exploration (exploring non-stationarity in a student's preferences) and exploitation (maximize a student's satisfaction over time)** [3]. For example, an adaptive teaching system should present different explanations knowing a student's preference for learning. However, unless we try different ways of teaching it is not possible to say with certainty whether or not an explanation would help a student learn effectively. We use the term adaptive teaching to avoid confusing it with adaptive learning used in machine learning literature. In the education domain, these terms are used interchangeably.

We represent this use case as a contextual bandit problem. We use contextual information about the student such as their preferences to learn through *visual, text, demo-based, practical, activity-based, step-by-step, lecture, audio-based explanations as well as self-evaluation and pre-assessment of students*. We also use contextual information about the content used to teach a topic, by rating them based on *ease of understanding, simplicity, intuitiveness, depth in teaching, conciseness, thoroughness, ratings, abstractness, hands-on, experimental*. A **content item or arms are different actions or ways a topic can be taught**. The reward would be the student's feedback to confirm their understanding of the topic they are trying to learn.

The feedback can be through quizzes, interactions with a content item, tasks to name a few. **By pulling an arm, we obtain a reward drawn from some unknown distribution determined by the selected content item and the context. Our goal is to maximize the total cumulative reward**.

Let us make this more concrete by mapping this use case to teaching a class. In any school, a course comprises of multiple topics. However now instead of a single way to teach everyone, there would be multiple ways to teach. These different ways to teach are called content items. Student's give their feedback on the presented content. Behind the scenes, our learning algorithm takes information about the student *(also referred to as student context)*, topic, content items*(also referred to as content context)* to find the best way to teach a student.

This project extends the most cited contextual bandit learning algorithm, LinUCB (Linear Upper Confidence Bound) [24] to enhance it for our use case. LinUCB was created by Yahoo! research to personalize news recommendation where the user's contextual data like IP address, history of prior visits, location among others. was used to personalize news recommendation. The feedback is in terms of click-through-rate (CTR). It is the number of clicks per news article. When a user clicks a news recommendation the learning algorithm receives a reward, or else they receive no reward. LinUCB increased CTR over Yahoo! Front Page Today Module dataset by 12.5% [24].

In some aspects, our use case is different from the news recommendation. Firstly, they have to recommend news from the vast pool of news articles. However, we represent adaptive teaching in a way that the learning algorithm has to select from content items relevant to the topic. Such a representation helps the algorithm learn quickly, which is necessary for adaptive teaching to keep students engaged with a system that understands their needs. News recommendation is a feature presented along with a host of other features. They can explore much longer. However, exploring excessively in adaptive teaching use case could leave students disillusioned with the system.

There are similarities between them. Both systems suffer from a cold-start situation and non-stationarity in users behavior. Cold start arises because a significant number of users are likely to be new with no historical information. Both use cases suffer from non-stationarity in user's behavior as preferences change with time. Hence, balancing exploration (to find content that matches the user's interest) and exploitation (present content that interest users) becomes critical for user satisfaction.

1.2 Motivation

The main problem with traditional education, which has been perpetual, is the enormous challenge teachers face for being responsible for ensuring every student can acquire expertise in their subject even though students may come from diverse backgrounds and interests [31]. In such classrooms learning has mostly remained a one-size-fits-all experience in which the teacher selects a learning resource for all students in their class regardless of their diversity in needs, understanding, ability, preferred learning style, and prior knowledge. It is not feasible for teachers to ensure their explanations can cater to all students. Hence there is a need for a system which could personalize teaching for students to help them learn effectively as well as increase course engagement and progression.

Such systems would be adaptive, recognize different levels of prior knowledge among students, as well as course progression based on a student's skill and feedback from learning. It would change teachers responsibility from a provider to a remediator and facilitator in teaching. These would adapt to individual student's learning patterns instead of a student having to adjust to the way of teaching. They would provide timely and comprehensive data-driven feedback to recognize potential challenges that students might come across as the course progresses.

1.3 Contribution

We present a novel baseline algorithm for our proposed adaptive teaching methodology which learns from students and contents for each topic to create a personalized learning path for every student. It adapts dynamically based on student's feedback and learning preferences.

We also provide a skip feature which is meant to keep a student engaged to increase their retention as well as provide feedback to teachers by recognizing the challenges faced by a student early in the course. Our online learning algorithm gives close to optimal results over a synthesized unbiased heterogeneous dataset.

1.4   Organization

Chapter 1 provided a brief overview of our use case along with the need for an adaptive teaching system and how this project contributes to realizing it. Chapter 2 introduces the technical concepts used to represent our use case along with the algorithm we customize for adaptive teaching. Chapter 3 describes prior work related to our use case using different approaches and how our work compares to them. Chapter 4 explains the algorithm created for adaptive teaching along with the skip feature. Chapter 5 provides the basis for the features selected and data synthesized to evaluate the algorithm. Chapter 6 describes the experimental setup along with the dataset synthesized to evaluate our algorithm. It also explains the evaluation strategy followed to examine our results. Chapter 7 presents the results of our experiments and compares our learning algorithm to the best possible policy. Chapter 8 concludes this project by summarizing the contributions and outlines possible avenues for future work.

Chapter 2

Preliminaries

This chapter briefly explains the key concepts used in this project.

2.1 Multi-armed bandit

Multi-armed bandit is a problem setting where an agent makes a sequence of decisions in time $1, 2, ..., T$. At each time t the agent is given a set of K arms to choose and has to pull an arm. As feedback for pulling an arm, it receives a reward for that arm, while the rewards of other arms cannot be determined. Problems represented as multi-armed bandits are stateless. They can be stochastic or adversarial. In a stochastic environment the reward of an arm is sampled from some unknown distribution, and in an adversarial setting the reward of an arm is picked by an adversary and may be sampled from a distribution [42]. In this project, we assume the problem setting as stochastic.

Personalized recommender systems recommend items (e.g., movies, news articles, web advertisements) to their users based on their predicted preference for these items. The feedback received from users response helps the system improve their prediction [2]. However, to improve the quality of recommendations, an item has to be recommended. If it is never recommended, the system cannot evaluate the feedback to improve the quality of predictions on these items. Such problems where there is some information about the users which can be used to improve the quality of predictions can be represented as a contextual bandit problem [40].

2.2 Contextual Bandit

In the theory of sequential decision-making, contextual bandit problems [37] are placed between multi-armed bandit problems [8] and full-blown reinforcement learning (usually modeled using Markov decision processes with discounted or average reward to take optimal actions) [35]. Traditional bandit algorithms do not use any side-information or context. However, contextual bandit algorithms use context to learn and map contexts to appropriate actions. Since bandit algorithms only have a single state they do not have to consider the impact of their actions over future states. Nevertheless, in many practical domains, such a problem setting is useful. It is true when the learner's action has limited impact on future contexts. In such a problem setting contextual bandit algorithms have shown great promise. Examples include web advertising [1] and personalized news article recommendation on web portals [24, 17].

Formally a contextual bandit problem is a repeated interaction which takes place over T rounds. At each round $t = 1, 2, ...T$ the environment reveals contexts $x_t \in X$ about the user and the available actions which are used by the learner to pick an action $a_t \in A$ which gives a reward r_t revealed by the environment. The goal of the learner is to choose an action which would maximize cumulative reward $\sum_{t=1}^{T} r_t$.

We will now translate this problem setting for our adaptive teaching use case in which an algorithm A which proceeds in discrete rounds $t = 1, 2, 3, ...$ In round t:

1. The algorithm observes the student context x_s and a set A_t of content items together with their feature vectors x_c for $a \in A_t$. X_t encapsulates x_s and the context x_c of all content items available in round t.

2. Based on observed rewards in previous rounds, A chooses an arm $a_t \in A_t$. The arm a_t is estimated to have the highest expected reward. In a stochastic setting, the expected reward is given as the inner product of an unknown arm-dependent parameter $\theta_{a,t}$ and the context $x_{t,a}$, that is, $E[r_{t,a}|\mathbf{x_{t,a}}] = x_{t,a}^T \theta_{\mathbf{t,a}}$.

3. The student reveals the received reward r_t for arm a_t whose expectation depends on both the context X_t and the arm a_t.

4. The algorithm then improves it's content item selection strategy with the new observation (x_t, a_t, r_t). It is important to emphasize here that no feedback namely, reward r_t is observed for unchosen arms $a \neq a_t$. This is valid for similar contextual bandit settings. [24].

2.3 Upper Confidence Bound (UCB)

An unavoidable challenge in bandit algorithms is to find the right balance between exploration and exploitation (Section 1.1). Upper Confidence Bound (UCB) comprises of a family of algorithms which try to find the best trade-off between exploration and exploitation. It is based on the principle of *being optimistic* by choosing actions which have the highest potential for reward. The reason this works is that when acting optimistically, one of two things happens. Either the optimism was well justified, in which case the learner is already acting optimally, or the optimism was not justified. In the latter case, the algorithm takes some action that they believed might give a reward when in fact it does not. If this happens sufficiently often, then the algorithm will learn the real reward of this action and not choose it in the future [23]. UCB algorithms estimate the expected reward for each arm by adding estimated sample mean of an arm with it's upper deviation.

We will refer to Figure 2.1 as an example to understand UCB. Let us assume we have three arms a_1, a_2, a_3. The reward distribution for each arm after several rounds is a Gaussian distribution Q with mean μ and standard deviation σ. The y-axis is the probability of obtaining a reward for these arms.
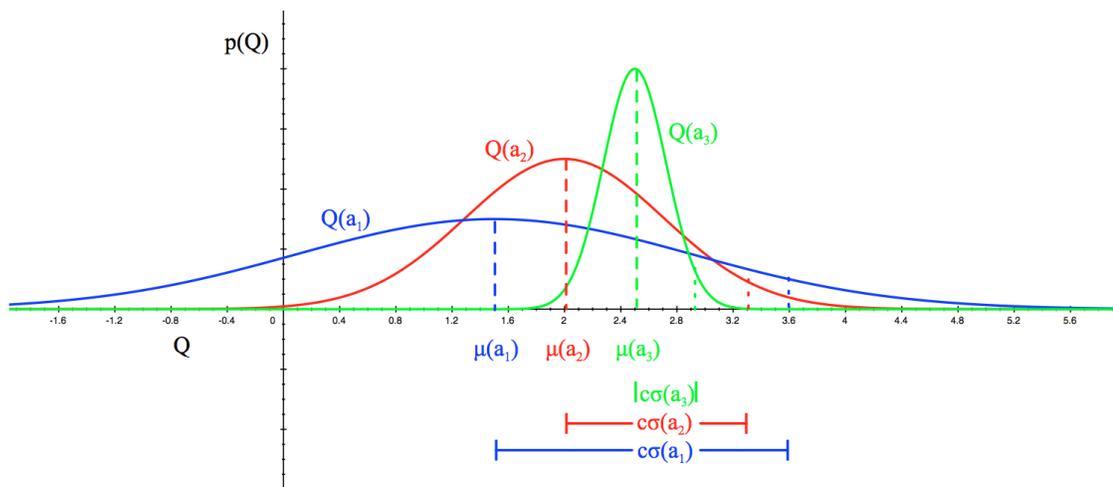
Figure 2.1: An example: UCB
[34]

The upper deviation for each arm is given by $c\sigma(a_i)$. The distribution shows that the sum of the expected mean and upper deviation is highest for a_1. Hence the UCB algorithm will select a_1. The reward received will reduce uncertainty around a_1. So

for the next round, the algorithm once again finds the arm with the highest sum for the expected mean and upper deviation. This is repeated for T rounds [10].

2.4   Linear Upper Confidence Bound (LinUCB)

LinUCB is a way to apply UCB to a more general contextual bandit setting where the UCB of each arm is computed efficiently by assuming the reward is linear, given as $E[r_{t,a}|\mathbf{x_{t,a}}] = x_{t,a}^T \theta_{\mathbf{t,a}}$. The estimated expected mean is parameterized over the context x_a for each arm a. At round t this is given as $\widehat{\theta}_a^T x_{t,a}$. The upper deviation around each arm a at round t is given as $\sqrt{x_{t,a}^T A_a^{-1} x_{t,a}}$. Here, A_a is the co-variance over the context data $x_{t,a}$ for each arm a at round t.

LinUCB introduces a hyper-parameter α, which allows us to control exploration over arms. This is achieved by scaling the upper deviation by α. A higher value of α encourages exploration. As a result, the algorithm would need more rounds to explore before it begins exploiting. We can now compute **the expected estimated reward for an arm** a **at round** t **as** $p_{t,a} = \widehat{\theta}_a^T x_{t,a} + \alpha \sqrt{x_{t,a}^T A_a^{-1} x_{t,a}}$ [24].

Chapter 3

Related Work

Our use case could also be represented using a partially observed Markov decision process (POMDP) framework. POMDPs model the student's latent knowledge states and their transitions to learn a policy that will present an action that could maximize reward received over the long run (long-term learning outcome). Previous work applying POMDPs to personalized learning has had limited success. However, to create a personalized learning schedule using a POMDP can get complicated and intractable as the number of dimensions representing the states and actions grows. As a consequence of this curse of dimensionality, POMDPs have had a limited impact to personalize learning in large-scale applications which has a large number of students and learning actions [22].

Reinforcement learning with Markov decision processes was used to define the learning path for each student. It represents the knowledge states a student can transition to [19]. At each state, a topic is presented to a student, and their knowledge is tested. The feedback received is used to update the usefulness of executing an action before the student transitions to the next state. A similar approach was applied in another system named ADVISOR [5]. However, it could not scale with the number of features. As the number of states increases, it becomes difficult for such a system to learn.

A more practical and tractable approach to personalized learning is to learn a policy, which maps contexts to actions using the multi-armed bandit (MAB) framework, which is more suitable for our use case. It makes it more practical than the MDP framework in large-scale educational applications [22].

eTutor [36] developed an online learning algorithm for personalized education. It uses the context of a student to decide the content item to present. It learns the

sequence to present content items to maximize the final score and reduce the time to teach. This system targets those who would like to refresh their knowledge. Our goal is to teach students who have little knowledge of the subject.

Multi-armed bandits have also been used to recommend courses to learners and assess their knowledge [29]. The works in [11, 21] both use expert knowledge to learn a teaching policy. Similar tasks are studies in [11], which uses domain expertise to reduce the set of possible actions a student can take. It is focused on assessments to test a students knowledge, using psychometrics paradigms such as Item Response theory and Zone of Proximal Development. Item response theory decides the content item to present to a student based on their response to previous 'n' content items. Zone of proximal development defines the content that should be presented to enhance a student's knowledge. It presents a content item that is just hard enough to challenge a student to keep them engaged. These studies do not focus on providing an adaptive learning path for navigating students to learn and understand new concepts. Our focus is on adaptive teaching, rather than adaptive testing.

The work in [25] applies a MAB algorithm to educational games to find a trade-off between exploring learning resources to accurately estimate arm means, while also trying to maximize users test performance. Their approach is context-free and does not consider diversity among individual users. The work in [27] collects data to find how students interact with the system to extract features as they play an educational game. It uses this knowledge to find a good teaching policy [22].

The work in [22] is focused on adaptive testing to assess students performance. They use contextual MAB to find questions to assess a student. The question depends on a student's response to earlier questions. At each round, they have all questions to assess a student. Contrary to that we only have a restricted set of content items available at each round. Our use case is focused on adaptive teaching to enable students to learn.

Other works typically create a model for each component, namely student, knowledge, domain and use knowledge tracing [13], item response theory and zone of proximal development [26, 32, 6] to make better decisions. These different methods have similar predictive performance. However, they could have very different teaching policies. [22]. While these results are different approaches to make the best prediction, none of them use machine learning to develop a policy learning algorithm.

Chapter 4

Algorithm

This chapter presents the algorithm created for the adaptive teaching system. We first present the basic version of the algorithm (Section 4.1). We then explain the skip feature (Section 4.2), which could streamline learning.

The algorithm used is an extension of upper confidence bound (UCB)-based algorithms [4] (Section 2.3). These algorithms maintain estimates of the expected reward of each arm together with confidence bound around it. It then **pulls the arm with the highest estimated reward which is equal to the sample mean plus the confidence bound**. Based on the actual reward it updates the arms parameters iteratively after each pull to make a better decision in upcoming rounds. In this project we are using the most cited contextual bandit algorithm, namely LinUCB (Section 2.4).

Before we dive in it is important to note, that to better understand the algorithm we have divided the explanation into two halves. *The first half explains the overall flow without skipping whereas the second explains in-depth the function calls made in the first half along with skipping.* We are using bandit terminology to explain. **Arm** refers to a content item. **Payoff** is the algorithms upwardly biased estimate of the expected reward, where the bias is due to the algorithms use of upper deviation rather than using the sample mean directly. A **round** comprises of computing the expected payoff for each content item; then presenting a content item with maximum expected payoff and getting student feedback for the content item.

Below are the notations used in the algorithm.

Symbol	Meaning
α	Parameter to scale Confidence bound.
C	Confidence threshold to skip.
$\mathbf{x_s}$	Student context vector.
x_c	Content items context matrix for a topic.
$\mathbf{x_t}$	Context vector at round t.
X_t / X_t^i	Context at round t. It combines $\mathbf{x_s}$ and all available x_c for topic i.
X_t^{i+1}	Context at round t. It combines $\mathbf{x_s}$ and all available x_c^{i+1} for topic $i+1$.
x_c^{i+1}	Content items contexts for topic $i+1$.
a	An arm a for topic i.
a'	An arm a' for topic $i+1$.
A_t	Arms available at round t.
$A_{t'}^{i+1}$	Arms available for topic $i+1$ at round t'.
a_t^{i+1}	Arm a for topic $i+1$ at round t.
t	Current round t.
t'	Possible next round t'.
i	Topic being taught.
$i+1$	Next Topic in the sequence.
$p_{t,a}$	Expected payoff from arm a at round t.
$p_{t,a}^i$	Expected payoff from arm a at round t for topic i.
$p_{t',a'}^{i+1}$	Expected payoff from arm a' at round t' for next topic $i+1$.
X	Input features for skip classifier.
Y	Label to train the skip classifier.

Table 4.1: Algorithmic notations

Note

- We are always on the current topic i, unless we explicitly specify next topic $i+1$.

- All vectors are **bold** faced lower cased.

- All sets are plain faced.

Algorithm 1 Teach with LinUCB

1: **Hyper Parameters :** $\alpha \in \mathbb{R}_+$
2: $\qquad\qquad\qquad\qquad C$: Confidence threshold to skip
3: **Inputs :** Student context $\mathbf{x_s}$ and content context x_c of available arm $a \in A_t$ for topic i at round t
4: Prepare context $X_t = \begin{pmatrix} \mathbf{x_s} \\ x_c \end{pmatrix}$
5: skip-enabled $\leftarrow$ False
6: **while** $A_t \neq \emptyset$ **do**
7: $\qquad a_t^i$, $p_{t,a}^i \leftarrow$ Expected-Payoff(X_t, A_t)
8: $\qquad$ skip-decision , $p_{t',a'}^{i+1} \leftarrow$ SkipTopic$(\mathbf{x_s}, p_{t,a}^i, i)$
9: $\qquad$ **if** skip-decision and skip-enabled is True **then**
10: $\qquad\qquad$ Move to next topic $i \leftarrow i + 1$
11: $\qquad\qquad$ **break**
12: $\qquad$ **else**
13: $\qquad\qquad$ Pull arm a_t and observe reward r_t
14: $\qquad\qquad A_{a_t} \leftarrow A_{a_t} + \mathbf{x_{t,a_t}} \mathbf{x_{t,a_t}^T}$
15: $\qquad\qquad \boldsymbol{b_{a_t}} \leftarrow \boldsymbol{b_{a_t}} + r_t \mathbf{x_{t,a_t}}$
16: $\qquad\qquad$ label $\leftarrow$ setlabel(r_t)
17: $\qquad\qquad$ Train$(\mathbf{x_s}, p_{t,a}^i, p_{t',a'}^{i+1}$,label$)$
18: $\qquad\qquad t \leftarrow t + 1$
19: $\qquad$ **if** $r_t \neq 1$ **then**
20: $\qquad\qquad$ Remove $a_t \in A_t$
21: $\qquad\qquad$ skip-enabled $\leftarrow$ True
22: $\qquad$ **else**
23: $\qquad\qquad$ Move to next topic : $i \leftarrow i + 1$
24: $\qquad\qquad$ **break**

4.1 Basic Version

The basic version is without skipping. It explains the main flow of the algorithm. The next section explains the functions used along with skipping.

The algorithm has hyper-parameters. The first one is α which scales the confidence bound (Section 2.4). The second hyper-parameter is the confidence threshold C which decides confidence threshold that must be exceeded to skip a topic. Skipping is a feature to help students who are unlikely to learn from content items available for a topic. It is meant to streamline learning. It could also be used by teachers to recognize topics students struggled on.

We now explain how LinUCB (Section 2.4) helps the algorithm decide an arm to pull. Before we recommend a content item to a student, we need to prepare context X_t for the round t. The context is a combination of the student context $\mathbf{x_s}$ and content items x_c for a topic i which the student is trying to understand. With the context X_t and arms A_t, we use LinUCB to compute the expected payoff from each arm and return the arm a_t^i with the maximum expected payoff $p_{t,a}^i$ which must be pulled for topic i at round t.

Assuming the classifier does not recommend skipping, a student is presented with the content item a_t for topic i. After being taught the student sends a reward r_t to complete the round t. Now the round t is complete we update the arm parameters A_{a_t} , $\mathbf{b_{a_t}}$ of the arm pulled. We then use this reward r_t to train the skip classifier to make better predictions in upcoming rounds. The features for the classifier comprise of a student's contextual information $\mathbf{x_s}$, expected payoff $p_{t,a}^i$ from the current topic i and the expected payoff $p_{t',a'}^{i+1}$ for the topic $i+1$.

A student $\mathbf{x_s}$ sends no reward r_t for a topic i it does not understand. In such a case the algorithm removes the presented arm a_t and remains on the same topic i. However, if a reward r_t is sent if a student understands a topic and is moved to the next topic $i+1$. This completes the first half. The second half explains the functions briefly described above.

4.2 With Skipping

On line 6 (Section 4.1) of the algorithm we get the expected payoff $p_{t,a}^i$ estimated on pulling the arm a_t^i for the current topic i. Now to decide whether or not it should pull the arm or move to the next topic it calls the skip topic function.

25: **function** SKIPTOPIC(x_s, $p_{t,a}^i$, i)

26:     Get next topic $i+1$ from topic i

27:     Get arms $A_{t'}^{i+1}$ and content context x_c^{i+1} for topic $i+1$

28:     Prepare context vector $X_{t'}^{i+1} = \begin{pmatrix} x_s \\ x_c^{i+1} \end{pmatrix}$

29:     $a_{t'}^{i+1}$, $p_{t',a'}^{i+1} \leftarrow$ EXPECTED-PAYOFF($X_{t'}^{i+1}$,$A_{t'}^{i+1}$)

30:     skip-decision $\leftarrow$ PREDICT(x_s, $p_{t,a}^i$, $p_{t',a'}^{i+1}$) to decide on skip

31:     **return** skip-decision , $p_{t',a'}^{i+1}$

32: **function** EXPECTED-PAYOFF(X_t,A_t)

33:     **for** $a \in A_t$ **do**

34:         Get $x_{t,a} \in X_t$

35:         **if** a is new **then**

36:             $A_a \leftarrow I_d$ (d-dimensional identify matrix)

37:             $\boldsymbol{b_a} \leftarrow 0_{d \times 1}$ (d-dimensional zero vector)

38:         $\widehat{\theta}_a \leftarrow A_a^{-1} b_a$

39:         $p_{t,a} \leftarrow \widehat{\theta}_a^T x_{t,a} + \alpha \sqrt{x_{t,a}^T A_a^{-1} x_{t,a}}$

40:     Choose arm $a_t = \arg \max_{a \in A_t} p_{t,a}$ with ties broken arbitrarily

41:     **return** $a_t, argmax p_{t,a},$

42: **function** PREDICT(x_s, $p_{t,a}^i$, $p_{t',a'}^{i+1}$)

43:     X $\leftarrow x_s$,i+1 ,$p_{t,a}^i$,$p_{t',a'}^{i+1}$

44:     Y , confidence-score $\leftarrow$ Prediction   from   classifier

45:     **if** confidence-score $<$C **then**

46:         decision $\leftarrow 0$

47:     **return**   decision , confidence-score

48: **function** TRAIN(x_s, $p_{t,a}^i$, $p_{t',a'}^{i+1}$, label)

49:     $X \leftarrow x_s, p_{t,a}^i, p_{t',a'}^{i+1}$, topic ,

50:     $Y \leftarrow$ label

51:     Train online SGD classifier

52: **function** SETLABEL(r_t)

53:     **if** r_t is 0 **then**

54:         $label \leftarrow 1$

55:     **else**

56:         $label \leftarrow 0$

57:     **return** label

The *SKIPTOPIC* function takes the student context $\mathbf{x_s}$, the expected payoff $p_{t,a}^i$ for pulling arm a at round t for topic i and the current topic i. It uses the topic i to get a reference to the next topic $i + 1$. Through the topic $i + 1$ it gets content items $A_{t'}^{i+1}$ and context data x_c^{i+1} associated those content items. After combining the contexts to prepare $X_{t'}^{i+1}$ it gets the maximum expected payoff $p_{t',a'}^{i+1}$ and the arm $a_{t'}^{i+1}$ to pull by passing the context vector $X_{t'}^{i+1}$ and arms available for next topic $A_{t'}^{i+1}$. The expected payoff function returns an arm with the maximum estimated payoff. Skip topic function then calls the skip classifier to predict a skip-decision for the student context $\mathbf{x_s}$, along with the expected payoff from the current and the next topic to make a prediction.

The *EXPECTED-PAY0FF* function takes the context X_t, along with the arms A_t available at round t. After an arm a_t is initialized with parameters A_a, b_a they are used to calculate the expected mean $\widehat{\theta}_a^T x_{t,a}$ and confidence bound $\sqrt{x_{t,a}^T A_a^{-1} x_{t,a}}$ for the arm. The confidence bound is scaled by α. The expected mean and the scaled confidence bound are added to give the expected payoff $p_{t,a}$ for arm a at round t. It then finds an arm a with maximum expected payoff $p_{t,a}$ and returns the expected payoff along with the arm a to be pulled.

The *PREDICT* function is used to predict whether the student should be moved to the next topic $i + 1$ or should remain on the same topic i. It combines student context vector $\mathbf{x_s}$, the expected payoff $p_{t,a}^i$ for the current topic i and the expected payoff $p_{t',a'}^{i+1}$ for the next topic $i + 1$ to prepare a feature vector X. It then gets a prediction from the binary supervised online support vector classifier with hinge loss to make a prediction Y, and a confidence-score for the prediction. If the confidence-score is less than the confidence threshold, then set the *decision* variable to 0 which implies no skipping. A confidence score lower than the threshold implies that the classifier is not sufficiently confident about the prediction.

The *TRAIN* function is used to train the skip classifier to make better predictions. Similar to the predict function it combines student context vector $\mathbf{x_s}$, the expected payoff $p_{t,a}^i$ for the current topic i and the expected payoff $p_{t',a'}^{i+1}$ for the next topic $i + 1$ to prepare a feature vector X. It sets the *label* to the output Y. Together they train the skip classifier.

The *SETLABEL* function is used to set the *label* to train the skip classifier. If the reward r_t for round t is set to 0 then the *label* is set to 1. It implies that since staying on the same topic did not give any reward; it would have been better to skip. If the reward r_t for round t is set to 1, then the *label* is set to 0. It implies that staying on

the same topic was a good decision. The *label* is used to train the skip classifier.

Chapter 5

Learning and Teaching

This chapter provides an overview of the data synthesized to train the learning algorithm. It begins by explaining the concept of learning styles and how initial research [16] proposed learning styles help students learn effectively (Section 5.1). However, recent research has disregarded this claim (Section 5.2). We then present how we use this knowledge in our project (Section 5.3.

5.1 Learning Styles

Growing classroom sizes have made it impossible for teachers to observe and teach in different ways consistently. Teachers have a momentous task of accommodating students from different backgrounds, ability levels, disabilities, interests, and motivation. It was essential to design a model that would improve teaching strategies to help students learn effectively [12].

Understanding how we learn would help teachers present information in an intuitive way. However, every person is unique in how they learn and have different levels of knowledge on the subject. A reliable model could empower teachers with a framework to prepare content items for their class. This lead to the birth of learning styles which are a group of ways in which people learn. These models categorized students based on how they absorbed information effectively. One such model is the VARK (Visual, Audio, Reading, Kinesthetics) model. It categorized students as either visual learners, who learn by seeing things or auditory learners who learn by listening to things or kinesthetic learners who learn by doing things such as engaging in physical activity. These studies led teachers to categorize their students into different groups

to customize their teaching methods for each group. Similar to VARK, there are at least 70 other learning styles. [12]

At first, research showed the efficacy of learning styles and how they could be pivotal to help students learn effectively [12, 16]. However, over time several scientific studies have found learning styles to be ineffective and challenged it [33, 14]. These studies found learning styles were not reliable and universally valid. In the next section, we will study why we cannot solely rely on learning styles.

5.2   Myth about Learning Styles

The theory and practice of learning styles have generated considerable interest and controversy over the past 20 years and more [12]. However, recent research has disregarded the importance of learning styles. Here are some essential reasons behind it.

We store information according to the meaning and not based on a specific learning style. For instance, people with better visual memory will be better at learning visually. Learning styles theories categorize students based on their style of learning. However, this has some limitations. Let us take a simple example that invalidates this claim. If a student wants to learn where Canada is on a map, it is always better to show it visually instead of explaining through audio. This is right for an auditory learner, kinesthetic or a visual learner.

To support the claim that people understand information based on meaning rather than a specific learning style an experiment was conducted where a chess board was set up and presented to novice and expert chess players for 5 seconds. It was found that novice players could only recollect a few correct positions of the pieces on a chessboard compared to experts who could remember almost the entire chess board. Supporters of learning style models such as VARK suggested that this was possible as experts had a better visual memory. Hence, another experiment was conducted. This time though the pieces were randomly configured on the chess board. This new configuration was shown to the experts. They could no longer recollect the correct positions of pieces on the chess board. The study showed that experts remembered the chess board when it was configured properly, i.e. the board has some meaning to them. However, when the board was randomly configured it had little meaning. Hence they could not recollect it [9].

Another reason why learning styles do not work is that **the best way to learn**

depends on the content itself. It implies that what we learn depends on how information is conveyed irrespective of the learning style. For instance, if a teacher wants to show how different songbirds look like, it has to show pictures or videos. That does not mean the student is a visual learner. The content was presented visually in this case. Same applies to the sounds made by these different birds. If we want the student to remember the sounds made by a bird, then it does not make them an auditory learner.

Lastly, **many things can be taught and learned using multiple senses**. Say, we want to teach basketball to a group of students. Well, there are multiple ways to do this. We can show them a game of basketball, so they could watch and learn the game. They can also learn by reading the rules of the game to learn how it is played. They could also listen to the commentary or play the game. So we could incorporate multiple sensory experiences into one to make it more meaningful. It is not that the visual learner would learn by only watching the game. They can also learn through other sensory means. Thus incorporating multiple senses makes learning more meaningful.

5.3   Use in Adaptive Teaching

In this project, we take student preferences along with the properties of content items as input to teach and provide a better learning experience. We present different content items and observe students feedback to make better decisions. We do not categorize students into learning groups.

We have used a subset of features to represent content items. These features convey information such as whether the content presented is theoretical or practical, presents surface level or in-depth knowledge of the topic among others. We believe that the content used to teach influences the learning experience.

It is important to note that we have prepared a dataset to represent the different ways, diverse set of students could be taught effectively. Our features are not binary (either 0 or 1). A binary feature would imply whether or not a student prefers a certain learning style. Instead, we recognize students have preferences and would like information to be conveyed in different ways. Hence, we record student preferences as a floating point number (between 0 and 1). The content presented has different properties in different proportions. For instance, a content item could have some level of theory, some practical exercises and have a certain level of depth in explaining the

topic. Hence, we represent content preferences as a floating point number (between 0 and 1). We have incorporated different properties of content to appeal to different students preferences to help them learn effectively.

Chapter 6

Experiments

This chapter explains the dataset (Section 6.1) used to evaluate the learning algorithm. It then describes the environmental setup (Section 6.2) used for these experiments. The next section explains how we evaluate our algorithm (Section 6.3) in absence of pre-existing benchmarks using an omniscient policy (Section 6.4). We then complete this chapter by briefly describing how the learning algorithm (Section 6.5) and the skip feature (Section 6.6) work in this experimental setup.

6.1 Dataset

Machine learning algorithms are data-driven. Due to the novelty of our approach to the best of our knowledge, there is no similar dataset available. Hence **we synthesize datasets to represent data generated by students taking courses in an adaptive teaching environment.**

An honest attempt is made to synthesize an unbiased dataset representative of the heterogeneous students and content items. Biased datasets tend to focus on targeted student groups (for instance, having many students who give positive feedback) and could result in skewed rewards. Contrary to this, our dataset is representative of diverse student and content data and is not skewed towards a particular student group or content type. **The contextual data is created by randomly sampling a uniform distribution U(0,1] to simulate the diverse student preferences and content features.**

6.1.1 Course

We use the following course for our experiments.

Course 3 : A course which has 100 topics. 400 students take it. There are 720 content items for 100 topics. So on an average, there are 7 content items per topic. We use this course to evaluate our algorithm's scalability.

6.1.2 Context

Students preferences collected through scientific experiments in which students are explained topics in different ways. These different ways represent diverse teaching methods and strategies used to recognize how students learn. Students are assessed to evaluate their understanding. Such an experiment would help us better understand different learning preferences and evaluate features of teaching material which helped them learn effectively.

Student data can be collected when a student interacts with the system to infer how they learn, what they struggle with and what helps them most. This information could be useful to tutor students.

Research has shown that students prefer to learn in various ways [15]. Though there is no unanimous consensus, there is a fair bit of research and understanding to support the different needs of a student. These needs are debatable with different opinions published targeting students from different streams, such as medical, law, management students. Reliably generalizing features of teaching material, methods and strategies is a different field of research. The features we consider are by no means exhaustive but a representative subset of the main features. The tables 6.1 and 6.2 describe the student and content context used for these experiments.

We assume there was a survey conducted among students to find how teaching could streamline learning? Student's gave their preferences on a scale of 1 to 10 with 1 being least preferred and 10 being most preferred. Although some features seem correlated, to avoid bias we do not consider it. With reliable and proven scientific conclusions these correlations could be considered in the future. The extent of correlations depends on the dataset. Correlations would simplify the dataset by reducing dimensionality and help the algorithm learn quickly. Without correlations, we got to evaluate the algorithm under the worst-case assumptions. We normalize these features between 0 to 1.

Student Context	Description
Visual (S_V)	How much preference is given to visual explanations (video, short-film, movie-clip, video blog's)?
Text (S_T)	How much preference is given to written explanations (books, articles, blogs, research papers)?
Demo-based (S_D)	How much preference is given to live experiments to help understand a concept?
Practical (S_P)	How much preference is given to an explanation, followed by a demo of the topic, and enabling students to perform it?
Step-by-step (S_S)	How much preference is given to a guide to practice, try and understand a topic in a systematic way?
Activity/Task-based (S_AT)	How much preference is given to content items which are interactive and require students to participate?
Lecture (S_L)	How much preference is given to being passive and listen to an expert explain the topic?
Audio (S_A)	How much preference is given to audio explanations (podcast, music)?
Self-evaluation (S_SE)	Students self-evaluate their readiness, motivation, excitement for the course.
Pre-assessment (S_PA)	Teachers conduct a pre-assessment of the pre-requisites required for the course.

Table 6.1: Student context

Below (Figure 6.1) is a student context data point which shows a student preference. It tells us that this student prefers visual (S_V), text(S_T), demo-based(S_D) methods of learning, but does not prefer practical (S_P), activity-based(S_AT), and did not fare well in the pre-assessment(S_PA). The student does not mind step-by-step(S_S), lectures(S_L) , audios(S_A) to learn and believes he/she is ready for the course (S_SE).

S_V	S_T	S_D	S_P	S_S	S_AT	S_L	S_A	S_SE	S_PA
0	0.87	0.82	0.88	0.36	0.6	0.06	0.66	0.56	0.66	0.07

Figure 6.1: Student context template

Content Context	Description
Ease of understanding (C_E)	How relatively easy is it to understand the content?
Simple/Intuition (C_I)	Does it provide a simple, intuitive understanding of the topic ?
Surface/In-depth (C_ID)	Does it provide a surface level or deep understanding of the topic?
Brief/Concise (C_C)	Is it short, to the point or descriptive, verbose and elaborative, keeping in mind that learners have different levels of concentration and capacity to remember?
Thorough (C_T)	How well does the content item cover the topic?
Preference/Well reviewed/Well rated(C_R)	How well rated is the explanation?
Theoretical/Abstract (C_A)	How theoretical or abstract is the content item?
Practical/Hands on (C_P)	Is it something that can be tried or experienced?
Experimental/Task-based (C_ETB)	Does it require a task to be completed to fully understand it, like collaboration with other students or some research/findings?

Table 6.2: Content context

Below (Figure 6.2) is a content context data point prepared for the course. This content item is thorough(C_T), practical(C_P), and experimentally sound(C_ETB), but not in-depth(C_ID),concise(C_C), and abstract(C_A). It is moderate in terms of understanding(C_E), intuitiveness(C_I) and has positive reviews(C_R).

	C_E	C_I	C_ID	C_C	C_T	C_R	C_A	C_P	C_ETB
C_1_1	0.45	0.72	0.31	0.05	0.91	0.75	0.06	0.88	0.97

Figure 6.2: Content context template

Apart from the above contextual data, there is a course which is taught. For our

experiments, we consider a typical course which comprises of topics to be taught. These topics are labeled as *T_1, T_2 ... T_25*. For e.g: T_1 refers to the first topic of the course. Each topic has between 5 to 20 different content items. Each content is labeled in the format *C_topic-id_content-number*. For e.g: C_1_2 refers to the second content item for topic T_1. We now have the required contextual information. Teacher outlines the sequence of topics for a course. Let us take an example to understand the data.

6.2 Environment

We simulate a course taken by students with an omniscient policy. We also simulate the same course taken by the same students with the learning algorithm. The omniscient policy and the learning algorithm decide the content item presented to each student. In a bandit setting, the content items are the arms. Several students take a course at the same time. Both the omniscient policy and the learning algorithm work in online mode. The learning algorithm updates it's parameters in each round to give better predictions.

6.3 Evaluation Strategy

Since there are no readily available benchmarks to compare our algorithm, we assume there exist an omniscient policy. This policy is optimized to recommend the best arm to pull.

We run the same course with an omniscient policy and the learning algorithm to evaluate our learning algorithm relative to the omniscient policy. The evaluation is conducted with and without skipping. Due to the stochastic nature of a student's feedback both the omniscient policy and the learning algorithm will run for a different number of rounds. However, the total cumulative reward available is the same for both of them. Hence we evaluate them based on cumulative reward accumulated over all rounds.

We simulate the student feedback as a Bernoulli distribution. Here, the probability of success is the maximum expected reward computed by the omniscient policy. This reward for an arm a with optimal parameters $\theta_{t,a}^*$ and with context vector $x_{t,a}$ at round t is given by $E[r_{t,a}|\mathbf{x_{t,a}}] = x_{t,a}^T \theta_{\mathbf{t,a}}^*$. It is the probability that the student would return a reward for a content item. Based on the reward received by the learning

algorithm the arm parameters are updated to make better decisions in the upcoming rounds. This experiment aims to find how well does our algorithm optimize an arm's parameters to match the omniscient policy.

6.4 Omniscient Policy

This policy knows all the probability distributions. At every step of makes the way it makes the best decisions as it knows the true distributions. It does not have to learn anything. It has optimal parameters θ^* for each arm. Hence it is expected to maximize the cumulative reward.

This policy calculates the expected payoff for each arm a available for a topic. It then selects the arm with the maximum expected payoff.

6.5 Learning Algorithm

The learning algorithm can adapt to several students at the same time to present a content item personalized for a student. For every topic, a student is trying to learn it gets the expected payoff for all available content items. It checks whether it should skip to next topic or remain on the current topic. Skipping is activated only when a student gives no reward for the presented content item.

When a student is on a topic, the algorithm presents a content item that could maximize rewards. After working through the content item, the student shares feedback on the content item. If it sends a reward, then it implies that the student understands the concept and moves to the next topic. If it sends no reward, then the student may be presented with the next best content item for the same topic or could be moved to the next topic in the course sequence.

Once a student shares feedback on a content item, the data is sent to train the skip classifier to make a better prediction in future rounds.

6.6 Skip Topic

The learning algorithm needs to checks with skip topic feature to decide if it should skip to the next topic. Skip topic predicts this by using a student's context along

with the estimated payoff for the current topic and the estimated payoff for the next topic in the course sequence.

It makes this decision using an online supervised learning stochastic gradient descent classifier with student context along with the estimated payoff of the current and next topic to make a decision. The label for the classifier depends on the reward received for the topic. If a reward is sent then the label is set to 0, or it's set to 1. The classifier makes its decision based on the feedback of all students to recognize topics and content items that students found difficult. It helps it learn to make a better decision.

Skip topic feature streamlines learning. If a student has been taught a topic once and was not satisfied with it, then there is an option to skip to the next topic or explain the same topic with a different content item. It skips topics that a student is unlikely to understand.

The skip classifier is a linear support vector machine estimator with hinge loss. The estimator is a regularized linear model with stochastic gradient descent (SGD) learning. The gradient of the loss is estimated, each sample at a time and the model updates along the way with a decreasing learning rate. The regularizer is a penalty added to the loss function that shrinks model parameters towards the zero vector using squared Euclidean norm [30].

Chapter 7

Results and Evaluation

This chapter presents results using the experimental set-up given in the previous chapter (Chapter 6). We evaluate the learning algorithm to the omniscient policy. Before evaluation we need to find optimal values for hyper-parameters α (Section 7.1) and confidence threshold C (Section 7.2). We then proceed to use these optimal values to evaluate the learning algorithm with and without the skip feature. (Section 7.3).

7.1 Confidence Bound α

Finding an optimal value for α is essential to optimize learning as it scales the confidence bound of each content item. An optimal value would enable the algorithm to balance exploration and exploitation. A higher value of α would imply the learning algorithm takes more rounds exploring which can lead to sub-optimal results.

This parameter is configured for the learning algorithm and not the omniscient policy. We empirically evaluated an optimal value for α the course (Section: 6.1.1). The graph (Figure 7.1) compares the number of rounds per reward ratio of the omniscient policy and the learning algorithm for different values of α. As expected, the omniscient policy needs the fewest rounds to per reward. For the learning algorithm $\alpha = 0.75$ and $\alpha = 2$ required the fewest number of rounds per reward. We have used $\alpha = 2$ for our experiments as it performs optimally with more exploration, compared to $\alpha = 0.75$
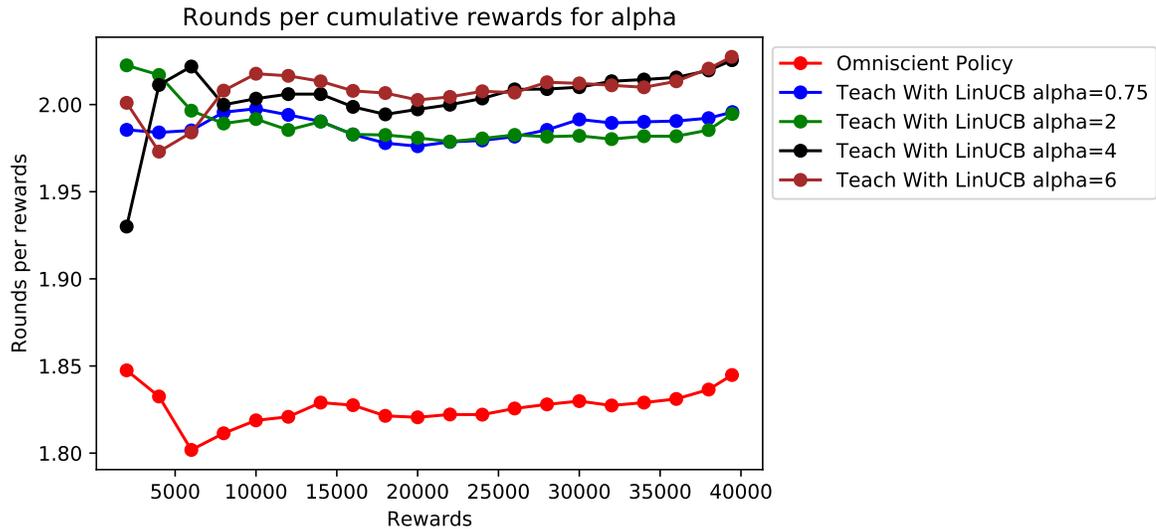
Figure 7.1: Rounds per cumulative reward ratio for α.

7.2 Confidence Threshold (C)

It is a threshold on the confidence score the skip classifier should exceed for its prediction to be accepted. Skipping is enabled for a topic only after a student gives no reward to a content item. The threshold helps:

- To keep a student engaged by skipping topics they are unable to understand.

- Give teachers control on their preference to skipping.

- Allow the learning algorithm to skip content items that are less likely to give rewards.

We do not want the confidence threshold to be too high as students might have to go through each content item nor do we want it to be too low such that students move to the next topic on the first occurrence of not understanding a topic. Hence finding an optimal value for the confidence threshold is essential to have a good learning experience.

We evaluate the performance for different values of the confidence threshold over course 1 (Section: 6.1.1). Below are the results.

7.2.1 Without confidence threshold

We evaluated the skip classifier with no confidence threshold. Below is a table that shows the results.

Reward per prediction type (in %).

Reward		Stay (0)	Skip (1)	Total
	0	25.10	18.28	43.38
	1	32.25	24.37	56.62

Table 7.1: Predictions without confidence threshold

We evaluate the classifier based on how well it helps the learning algorithm maximize cumulative reward. We found it to be accurate by **56.62%**. Accuracy is a metric to show how well it helped increase cumulative reward.

7.2.2 With confidence threshold

We evaluate the skip classifier with confidence threshold. We will only consider data points where the classifier's decision is overruled as its confidence score was below the threshold. It would be when the classifier had predicted skipping to the next topic, but since the confidence score was below the threshold, the prediction was ignored. It gives us a measure of effectiveness for the confidence threshold.

We evaluated the classifier for different values of confidence threshold. For different threshold values performance ranged consistently between 56 - 60 %. We found the skip classifier performed most optimally when the confidence threshold is 30. The below table 7.2 shows the results.

Reward per prediction type (in %).

Reward		Stay (0)	Skip (1)	Total
	0	18.3	24.18	42.48
	1	18.3	39.22	57.52

Table 7.2: Predictions with confidence threshold of 10

The above table shows us that by **57.52%** it took the correct decision. This helped increase the cumulative reward. As the value of the confidence threshold is increased the number of skips decrease. Table 7.3 shows the results for confidence threshold of 30.

Reward per prediction type (in %).

		Stay (0)	Skip (1)	Total
Reward	0	36.82	4.09	40.91
	1	50.45	8.64	59.09

Table 7.3: Predictions with confidence threshold of 30

The above table shows us that it made it the correct decision by **59.09%** which helped to increase the cumulative reward.

7.3   Learning Algorithm

We now evaluate the learning algorithm with and without the skip feature.

7.3.1   Without Skipping

With skipping disabled the only way a student can move to the next topic is by understanding it or until all content items have failed to explain the student. It increases the number of rounds required by a student to complete a course.

The graph (Figure 7.2) shows the cumulative reward of the learning algorithm and the omniscient policy. The reward for the omniscient policy increases linearly, whereas that of the learning algorithm is similar to the optimal policy. We expect this as we do not have the optimal arm parameters pre-configured. The algorithm learns the optimal parameters in each round.
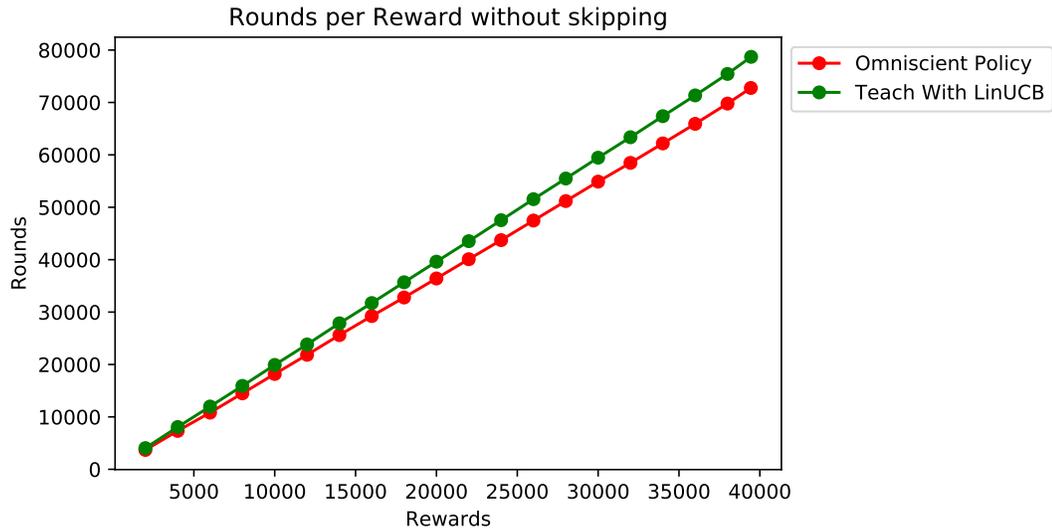
Figure 7.2: Rounds per reward without skipping.

The omniscient policy required 72,767 rounds to get a cumulative reward of 39,444. It needs 1.84 rounds for a reward (of 1). For the same course, the learning algorithm required 78,707 rounds to get a reward of 39,460. It needs 1.99 rounds for a reward (of 1). The graph (Figure 7.3) shows the number of rounds per reward required by the algorithm at different intervals with optimal values of hyper-parameters and compares it to the omniscient policy. It shows that our learning algorithm reduces the number of rounds per reward. This shows that our algorithm is learning in each round.
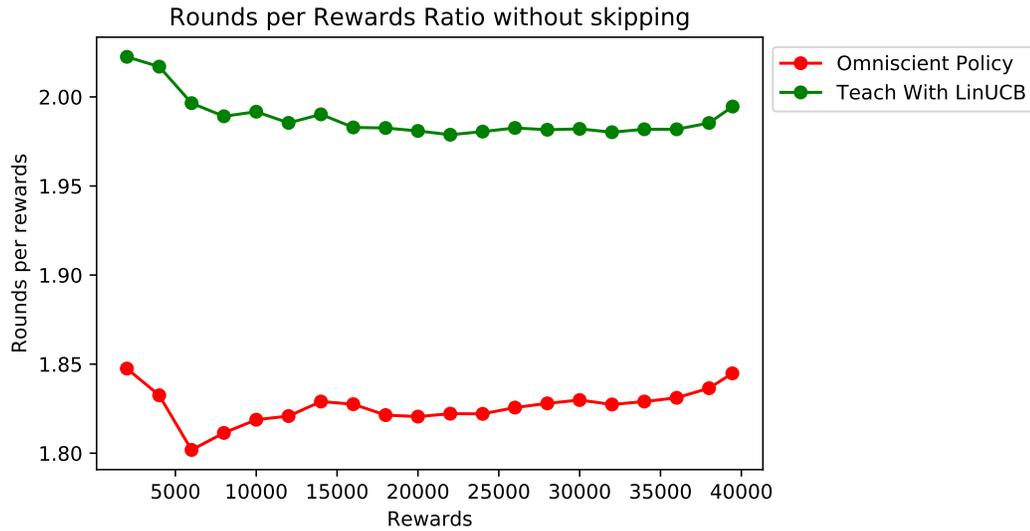
Figure 7.3: Rounds per rewards ratio without skipping.

7.3.2   With Skipping

If a student does not understand a topic then skipping is enabled. It does not imply the student is taken to the next topic. For that to happen, the skip classifier should be confident beyond the confidence threshold to predict that it would be better to take the student to the next topic.

Skipping tells the learning algorithm to skip sub-optimal content items and instead move to content items that have a higher estimated reward. It ensures that we do not present content items which are unlikely to help a student understand the topic. The graph (Figure 7.4) shows results of the learning algorithm with optimal confidence threshold $C = 30$ and $\alpha = 2.0$.
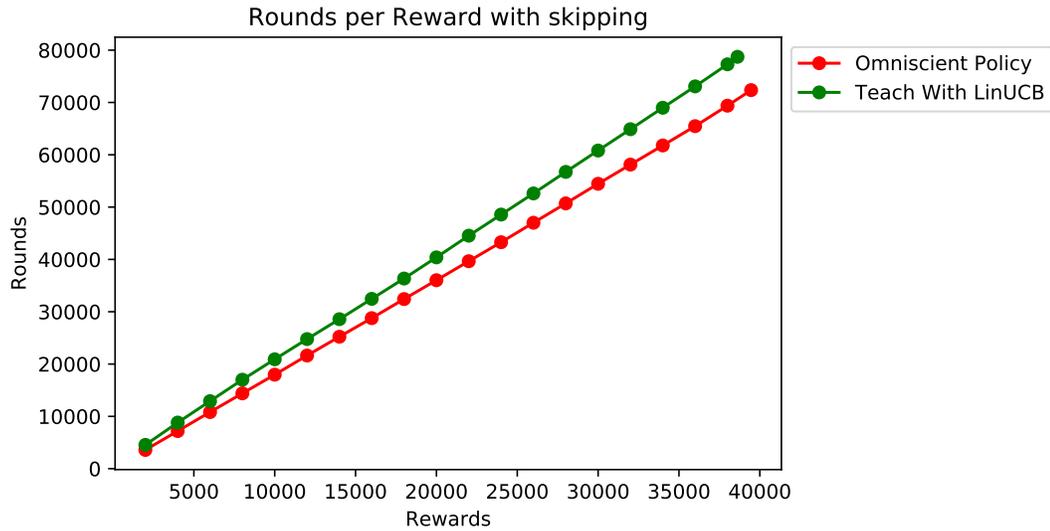
Figure 7.4: Rounds per reward with skipping

The graph (Figure 7.4) shows the performance of the learning algorithm and the omniscient policy. The number of rounds and the cumulative reward reduces with skipping enabled. The cumulative reward reduces as for topics that a student did not understand. These are instances where the skip classifier predicted with high confidence that it would be better to take the student to the next topic.

The omniscient policy required 71,449 rounds to get a cumulative reward of 38,891. It needs 1.84 rounds for a reward (of 1). The learning algorithm required 77,179 rounds to get a reward of 37,833. It needs 2.04 rounds for a reward (of 1). The graph (Figure 7.5) shows the number of rounds per reward required by the algorithm at different intervals for optimal values of hyper-parameters and compares it to the omniscient policy. It shows that the number of rounds per reward required by the learning algorithm reduces as the course progresses.
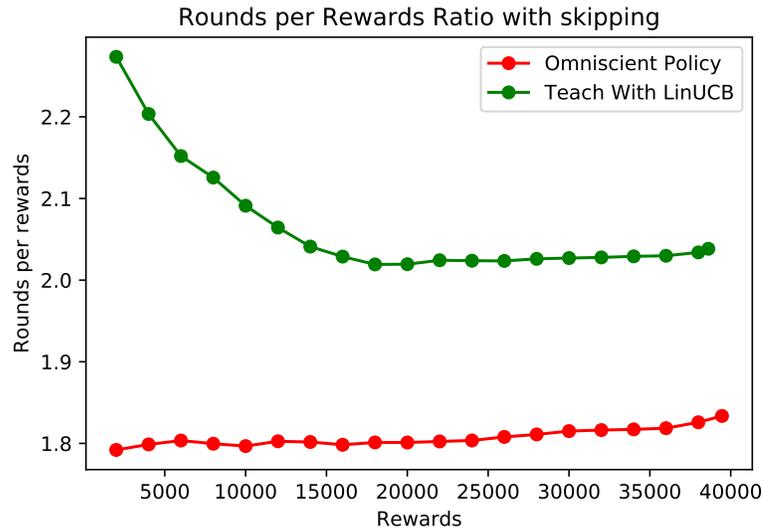
Figure 7.5: Rounds per reward ratio with skipping.

Comparing the cumulative reward graph with and without skipping shows us that without skipping our learning algorithm requires fewer rounds per reward than with skipping. However, without skipping it needs more rounds which could affect student experience.

7.4 Summary

In this chapter, we tuned our hyper-parameters α and confidence threshold C to evaluate our learning algorithm. We found that the learner performs better without skipping. However, skipping is required to improve a students experience.

In future works, we shall optimize the confidence threshold to decay over time to give better results. Furthermore, warm starting the skip classifier should also improve it's performance. In the current setup, we do not penalize incorrect predictions made by the learning algorithm. More research is required to find if penalizing predictions which gave no rewards would help improve the learner.

Chapter 8

Conclusions

This project presents a student-centric approach to teaching. An approach which could make a classroom more interactive by providing a personalized learning experience for students. We synthesized an unbiased dataset to represent heterogeneous student and content data to evaluate our learning algorithm. Since there is no benchmark available, we created an omniscient policy which has optimal parameters preconfigured. The algorithm learns these parameters to find an optimal content item for each student.

We then present a feature which would be useful when there are several different content items for a topic to avoid a student from getting frustrated by being unable to understand a topic. It helps not only students but also teachers to recognize topics students are less likely to understand. We evaluated the learning algorithm to set a baseline for this new teaching methodology.

Our future work would involve creating an actual course that follows the teaching methods outlined in this project. It would give real-world student data to evaluate the algorithm. We would also like to design other algorithms to evaluate their performance against our baseline algorithm. An additional optimization would be to find an optimal strategy to introduce skipping such that it does not restrict exploration and still provides an excellent student experience.

Bibliography

[1] Naoki Abe and Atsuyoshi Nakamura. Learning to optimally schedule internet banner advertisements. In *ICML*, volume 99, pages 12–21, 1999.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, (6):734–749, 2005.

[3] Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, Nitin Motgi, Seung-Taek Park, Raghu Ramakrishnan, Scott Roy, and Joe Zachariah. Online models for content optimization. In *Advances in Neural Information Processing Systems*, pages 17–24, 2009.

[4] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[5] Joseph Beck, Beverly Park Woolf, and Carole R Beal. Advisor: A machine learning architecture for intelligent tutor construction. *AAAI/IAAI*, 2000(552-557):1–2, 2000.

[6] Yoav Bergner, Stefan Droschler, Gerd Kortemeyer, Saif Rayyan, Daniel Seaton, and David E Pritchard. Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. *International Educational Data Mining Society*, 2012.

[7] Peter Brusilovsky and Christoph Peylo. Adaptive and intelligent web-based educational systems. *International Journal of Artificial Intelligence in Education (IJAIED)*, 13:159–172, 2003.

[8] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[9] William G Chase and Herbert A Simon. Perception in chess. *Cognitive psychology*, 4(1):55–81, 1973.

[10] Ankit Choudhary. Reinforcement learning guide: Solving the multi-armed bandit problem from scratch in python, September 24, 2018.

[11] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, and Manuel Lopes. Multi-armed bandits for intelligent tutoring systems. *arXiv preprint arXiv:1310.3174*, 2013.

[12] Frank Coffield, David Moseley, Elaine Hall, Kathryn Ecclestone, et al. Learning styles and pedagogy in post-16 learning: A systematic and critical review, 2004.

[13] Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[14] Lynn Curry. A critique of the research on learning styles. *Educational leadership*, 48(2):50–56, 1990.

[15] Richard M Felder. Matters of style. *ASEE prism*, 6(4):18–23, 1996.

[16] Richard M Felder, Linda K Silverman, et al. Learning and teaching styles in engineering education. *Engineering education*, 78(7):674–681, 1988.

[17] Kristjan Greenewald, Ambuj Tewari, Susan Murphy, and Predag Klasnja. Action centered contextual bandits. In *Advances in neural information processing systems*, pages 5977–5985, 2017.

[18] Trude Heift and Devlan Nicholson. Web delivery of adaptive and interactive language tutoring. *International Journal of Artificial Intelligence in Education*, 12(4):310–325, 2001.

[19] Ana Iglesias, Paloma Martínez, Ricardo Aler, and Fernando Fernández. Learning teaching strategies in an adaptive and intelligent educational system through reinforcement learning. *Applied Intelligence*, 31(1):89–106, 2009.

[20] Kenneth R Koedinger, John R Anderson, William H Hadley, and Mary A Mark. Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education (IJAIED)*, 8:30–43, 1997.

[21] Kenneth R Koedinger, Emma Brunskill, Ryan SJd Baker, Elizabeth A McLaughlin, and John Stamper. New potentials for data-driven intelligent tutoring system development and optimization. *AI Magazine*, 34(3):27–41, 2013.

[22] Andrew S Lan and Richard G Baraniuk. A contextual bandits framework for personalized learning action selection. In *Educational Data Mining*, pages 424–429, 2016.

[23] Tor Lattimore. The upper confidence bound algorithm, September 18, 2016.

[24] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[25] Yun-En Liu, Travis Mandel, Emma Brunskill, and Zoran Popovic. Trading off scientific knowledge and user learning with multi-armed bandits. In *Educational Data Mining*, pages 161–168, 2014.

[26] Frederic M Lord. *Applications of item response theory to practical testing problems*. Routledge, 2012.

[27] Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1077–1084. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[28] Antonija Mitrovic. An intelligent sql tutor on the web. *International Journal of Artificial Intelligence in Education*, 13(2-4):173–197, 2003.

[29] Minh-Quan Nguyen and Paul Bourgine. Multi-armed bandit problem and its applications in intelligent tutoring systems. *Master's thesis. École Polytechnique*, 2014.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[31] Lou Pugliese. Adaptive learning systems: Surviving the storm, October 17, 2016.

[32] Mark D Reckase. Multidimensional item response theory models. In *Multidimensional Item Response Theory*, pages 79–112. Springer, 2009.

[33] Cedar Riener and Daniel Willingham. The myth of learning styles. *Change: The magazine of higher learning*, 42(5):32–35, 2010.

[34] David Silver. Exploration and exploitation.

[35] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[36] Cem Tekin, Jonas Braun, and Mihaela van der Schaar. etutor: Online learning for personalized education. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5545–5549. IEEE, 2015.

[37] Ambuj Tewari and Susan A Murphy. From ads to interventions: Contextual bandits in mobile health. In *Mobile Health*, pages 495–517. Springer, 2017.

[38] Rice University. Openstax tutor, 2016.

[39] Kurt Vanlehn, Collin Lynch, Kay Schulze, Joel A Shapiro, Robert Shelby, Linwood Taylor, Don Treacy, Anders Weinstein, and Mary Wintersgill. The andes physics tutoring system: Lessons learned. *International Journal of Artificial Intelligence in Education*, 15(3):147–204, 2005.

[40] Joannes Vermorel and Mehryar Mohri. Multi-armed bandit algorithms and empirical evaluation. In *European conference on machine learning*, pages 437–448. Springer, 2005.

[41] Beverly Park Woolf. *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann, 2010.

[42] Li Zhou. A survey on contextual multi-armed bandits. *arXiv preprint arXiv:1508.03326*, 2015.