Scholars' Mine

Masters Theses | Student Theses and Dissertations

Summer 2020

On predicting stopping time of human sequential decision-making using discounted satisficing heuristic

Mounica Devaguptapu

Follow this and additional works at: https://scholarsmine.mst.edu/masters_theses

Part of the Computer Sciences Commons

Department:

Recommended Citation

Devaguptapu, Mounica, "On predicting stopping time of human sequential decision-making using discounted satisficing heuristic" (2020). *Masters Theses*. 7951.
https://scholarsmine.mst.edu/masters_theses/7951

This thesis is brought to you by Scholars' Mine, a service of the Missouri S&T Library and Learning Resources. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

ON PREDICTING STOPPING TIME OF HUMAN SEQUENTIAL DECISION-MAKING USING DISCOUNTED SATISFICING HEURISTIC

by

MOUNICA DEVAGUPTAPU

A THESIS

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

in

COMPUTER SCIENCE

2020

Approved by:

Dr. Venkata Sriram Siddhardh Nadendla, Advisor
Dr. Donald Wunsch
Dr. Tony Luo

Copyright 2020

MOUNICA DEVAGUPTAPU

All Rights Reserved

ABSTRACT

Human sequential decision-making involves two essential questions: (i) "what to choose next?", and (ii) "when to stop?". Assuming that the human agents choose an alternative according to their preference order, our goal is to model and learn how human agents choose their stopping time while making sequential decisions. In contrary to traditional assumptions in the literature regarding how humans exhibit satisficing behavior on instantaneous utilities, we assume that humans employ a *discounted satisficing* heuristic to compute their stopping time, i.e., the human agent stops working if the total accumulated utility goes beyond a dynamic threshold that gets discounted with time. In this thesis, we model the stopping time in 3 scenarios where the payoff of the human worker is assumed as (i) single-attribute utility, (ii) multi-attribute utility with known weights, and (iii) multi-attribute utility with unknown weights. We propose algorithms to estimate the model parameters followed by predicting the stopping time in all three scenarios and present the simulation results to demonstrate the error performance. Simulation results are presented to demonstrate the convergence of prediction error of stopping time, in spite of the fact that model parameters converge to biased estimates. This observation is later justified using an illustrative example to show that there are multiple discounted satisficing models that explain the same stopping time decision. A novel web application is also developed to emulate a crowd-sourcing platform in our lab to capture multi-attribute information regarding the task in order to perform validations of the proposed algorithms on real data.

ACKNOWLEDGMENTS

Firstly, I would like to thank my advisor Dr. Venkata Sriram Siddhardh Nadendla, without whom I would not take up the research in the first place. Dr. Sid consistently supported and encouraged me and was there just a call away whenever I hit a wall. I would like to thank Dr. Donald Wunsch and Dr. Tony Luo for accepting my request to be a part of my thesis committee and for providing their valuable insights on my work. I would also like to take this opportunity to thank my fellow lab students at CPHS Laboratory for looking at my problem in a new perspective and raising questions that helped me to improve my work. Last but never least, I would like to thank my sister, parents, and friends for their never-ending support, prayers, and love.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

Figure Page

LIST OF TABLES

1. INTRODUCTION

Sequential decision-making is a situation where an agent makes a series of decisions over multiple stages. At each such decision stage, the agent observes all the available alternatives and chooses one that it believes is the optimal choice to meet its final expected outcome. Here an agent can be a human or a computer algorithm. As computer scientists, we are more interested in the designing algorithmic agents that make optimal decisions when presented with a situation where sequential choices have to be made. However, all the decisions/choices might not encompass an immediate or a deterministic reward, and such delays/uncertainties in the outcome make agents rely on the feedback while making their next decision. This uncertainty leads to two major questions: (i) "what to choose in the next stage?", and (ii) "when to stop?". These problems have been studied extensively in the literature and many have modeled real-world problems/situations as sequential decision-making problems and designed algorithms to find the optimal choice and the optimal stopping time. For example, Roohnavazfar *et.al.*, in [1], modeled the problem of searching optimal path in a dynamic network as a sequential decision-making problem and proposed a solution to find an optimal route using Nested Multinomial Logit model. Wu *et.al.*, in [2], modeled dynamic rate allocation and spectrum sharing problem in cognitive radio networks as a sequential decision problem and proposed an algorithm to minimize the average total power consumption in each scheduling cycle using dynamic programming. Schulze *et.al.*, in [3], explained how a worker sequentially chooses tasks in a crowdsourcing platform like Amazon MTurk. In crowdsourcing applications, researchers have also designed many recommendation engines to assist worker's decisions [4, 5].

Note that all the works mentioned earlier modeled agents as expected utility maximizers, where the agents' objective is to minimize cost, or maximize reward. In fact, most of sequential choice settings addressed in the literature rely heavily on the theory of expected utility maximization (EUM), which is constructed over an axiomatic framework proposed by Von Neumann and Oskar Morgenstern [6]. It has been shown through many psychological experiments that human abilities have both physical as well as cognitive limitations [7]. Such limitations naturally violate some of the axioms in EUM theory, which makes human decision makers incapable of performing an indefinite search for the optimal solution. Simon first pointed out these violations in his seminal book on Bound Rationality [8] and introduced the term *Satisficing* to characterize human decision-making. He defined satisficing as an agent's decision to stop choosing among the options when a threshold is met. However, in this thesis, we propose a novel heuristic called *Discounted Satisficing* to model the diminishing zeal due to increasing lag in deciding the stopping time. This diminishing zeal can be mathematically modeled as a discount factor that deteriorates the decision maker's threshold, and the agent avoids making a decision until the discounted threshold is met.

1.1. MOTIVATION FOR THIS WORK

Discounted satisficing heuristic models our ability to perform mental tasks and make decisions wears thin when it's repeatedly exerted. For example, in [9], Danziger *et al.* showed that prisoners prefer to have parole approved in the morning than in the afternoon as they believe that the judge's decision might be affected by the number of cases he/she has heard all day. Likewise, many people strongly believe that "More isn't always better." According to a study conducted by Sheena Iyenger and Mark Lepper in 2000 [10], the popular belief in retail markets that a customer is more likely to find the right product if given more options is asserted to be wrong. In fact, the paper demonstrated that the table presenting more options of jam for customers to try is less likely to being purchased from. This is because people often experience fatigue when presented with multiple options to choose from or when the outcome of the decision is not immediately observed. In such situations, people make decisions hastily, which are later repented and can have a severe effect on their mental and physical health. For instance, to guard innocent online users against bullying, fake news, disturbing and even illegal content on their social media handles, companies employ content moderators who carefully examine the user-uploaded content on the company's social media and remove them whenever they are deemed against the company guidelines. Companies like Facebook, Twitter, Youtube, and Instagram, whose main business is into social media, have millions of users uploading diverse content on their social media handles. An average content moderator reviews thousands of videos, tweets, pictures each day dealing with extreme content taking a mental toll on them. A recent article in *The Verge* [11] reported that the tech giant *Facebook* offered a settlement of $52 million to more than 11,000 content moderators who developed depression, addictions, and other mental health issues while they worked moderating content on their social media platform. Facebook outsourced this job to *Cognizant*, which employs most of its workforce in India and the U.S. to perform content moderation. When Verge interviewed 1,000 content moderators at Cognizant's Phoenix site, most of them reported an overpowering urge to sob and increased feelings of isolation and anxiety. To mitigate this problem, many companies used automated tools based on Artificial Intelligence (AI) such as object recognition algorithms to identify objectionable content within images [12]. However, this automated approach has not been very effective for diverse reasons such as heterogeneous content (e.g. images, videos, hate messages and tweets), multi-attribute sensitivity (e.g. race, gender and color) and contextual information. As a solution, *Accenture* recently proposed a 'bionic' content moderation model, a human-in-loop AI system [13], which decreases the need for thousands of human content moderators via automating the flagging process using AI algorithms in order to decrease the number of review tasks. Another recent work done at Google [14], discusses how AI can be used in reducing emotional impact on the content moderator via gray-scaling and blurring images.

In spite of all these attempts, the problem of cognitive distress amongst content moderators continues to linger within this industry. In fact, this problem also exists in many other sectors where workers perform multiple tasks sequentially for long hours without breaks. For example, the most common problem faced by an assembly worker in a manufacturing production line is the lack of flexible breaks whenever needed. Likewise, driving fatigue is the cause for $\sim 13\%$ of all large-truck accidents in the U.S. each year. In the same way, work-related post-traumatic disorder is common in military and police professionals, health care workers, firefighters, and first responders. The common reason behind all of the aforementioned cases is that people make ineffective decisions regarding when to stop working because of their financial problems, low economic background and performance pressure within the organization. Consequently, these workers pay a high price via compromising their mental health in the long run.

1.2. OUR CONTRIBUTION

In this thesis, my goal is to predict the worker's stopping time via modeling their decisions using a discounted satisficing heuristic. By doing so, our vision is to lead the research community towards mitigating mental stress disorders through effective interventions that are designed based on the worker's cognitive state. The main contributions of this thesis are three-fold:

- model worker's sequential decisions using discounted satisficing heuristic
- develop novel learning algorithms to estimate model parameters,
- validate these models using simulation and real-world experiments.

More specifically, the stopping time of a human agent is examined in 3 different scenarios by modeling his/her utility as (i) a single-attribute function, (ii) a multi-attribute function with known weights (iii) a multi-attribute function with unknown weights. Algorithms have been developed to estimate model parameters and predict stopping time efficiently in all three scenarios. The first algorithm is designed to estimate model parameters using concepts such as log-linearization, linear algebra, and polygon clipping, when agent decisions are based on a single-attribute utility. Although the algorithm has shown good convergence on the simulation-data, the performance of the algorithm on real-data is not very encouraging. This poor performance has motivated us to model agent decisions using a multi-attribute utility, for which three algorithms (two online and one batch learning) have been proposed in the case of known weights and one in the case of unknown weights to estimate the model parameters. These algorithms are designed using quadratic programming and alternating minimization techniques, and have shown a good convergence rate in the stopping time prediction on the simulation data.

The rest of the document is organized as follows. Section 2 provides an overview of the related work in human sequential decision-making. Section 3 discusses the proposed discounted satisficing heuristic in detail, along with proposed algorithms designed to estimate model parameters. Section 4 provides the simulation results and overview of the crowdsourcing platform developed to carry out validations on real-data. Finally, Section 5 summarizes our work in this thesis and provides an insight into our future work.

2. LITERATURE REVIEW

Sequential decision-making in human agents has been extensively studied by diverse researchers from varied fields such as neuroscience [15, 16], computer science [1, 3, 2], statistics [17] and psychology [18, 19]. While computer scientists and statisticians focused on finding the optimal stopping time, neuroscientists and psychologists showed interest in modeling rationality behind human sequential decision-making. Various models have been proposed to describe the sequential decision-making in humans, most of which agree upon a behavior that people experience an exploration/exploitation dilemma while making an action/decision. For example, a person hunting for the best possible apartment in New York City may not know if the apartment they see next is the best for them, without any knowledge of all the unknown choices. This dilemma induces a trade-off between choosing an option via exploiting current knowledge (e.g. accept the apartment they currently like) and deciding to explore other options to learn the best choice (e.g. keep searching for new apartments). However, people's limitations (both cognitive and physical) makes it impossible to pursue an indefinite search for the best solution, thus making them to settle for a sub-optimal solution. This problem of finding an optimal solution is mathematically referred to as "Optimal Stopping Time Problems".

Before going deep into the optimal stopping time problems, let us first understand different paradigms/frameworks available in the literature to model the human sequential decision-making.

2.1. MULTI-ARMED BANDIT FRAMEWORK

The most used framework to model human sequential decision-making is *Multi-Armed Bandit Framework*. Thompson first coined this framework in 1933 for the application of clinical trials where he is trying to prescribe the treatment that has a higher success rate among the two available experimental treatments to as many patients as possible [20]. However, the effectiveness of the two available experimental treatments is unknown. Hence, choosing the treatment is decided sequentially based on past patients' responses to his prescribed treatment. Such sequential decision settings are mathematically modeled as multi-armed bandit framework where an agent (human) is given multiple alternatives at each discrete-time, and when he/she chooses an alternative, observes a reward drawn from unknown probability distribution associated with each alternative. This framework first appeared in the literature in the late 1950s and early 1960s [21, 22, 23] as a reward-maximization problem where the agent plays a bandit machine with multiple arms, each of which, when played, yields a random reward drawn from an unknown distribution. This framework can be practically applied in many real-world situations such as routing algorithms in a communication network, tuning the look and feel of a website, datacenter design, radio networks. While dynamic programming offers a general solution for obtaining the optimal solution, its complexity grows exponentially with the number of alternatives. A practical solution was proposed to this computationally challenging problem in 1974 when Gittins and Jones [24] proposed a dynamic allocation index, where they proved that selecting the option with the highest index results in an optimal solution. However, the dynamic allocation algorithm suffered from two drawbacks incase of Restless Bandits (where multiple bandits can be chosen at a discrete time), (i) computational difficulty and (ii) lack of insight into the nature of optimal policies. Addressing these drawbacks, there are techniques developed to improve the human sequential decision-making using Reinforcement Learning algorithms such as Q-learning, SARSA [25, 26, 27, 28], Bayesian Modeling [29, 30], differential equations [31] and decision tree with branch-to-branch interactions [32].

2.2. OPTIMAL STOPPING TIME

Having known the optimal choice, the next important question in human sequential decision-making is: *when to stop?*. The theory of optimal stopping (or) early stopping is concerned with choosing a time to take a particular decision that involves whether to continue the experiment or quit. This problem is not new in mathematics literature [33] and has been studied in a gamut of domains such as statistics, economics, finance, and science (e.g. secretary problem, parking search problem, house selling problem [34]). This theory can address a range of issues, especially in the computer science domain, such as designing recommendation systems [35], enabling the caching in a server to accurately handle the object refreshing and the stale delivery problem [36], and deciding the stopping time for software testing [37]. Another interesting application lies in the area of computational finance, where online sellers use dynamic pricing by learning the purchase patterns of the consumers. Very recent work used a stopping time approach to protect consumers revealing their interests to these algorithms [38]. Traditionally, all the algorithms designed to predict the optimal stopping time models the decision maker as an Expected Utility maximizer who either attempts to maximize reward, or minimize cost.

2.3. EXPECTED UTILITY MAXIMIZATION (EUM) THEORY

The concept of expected utility is first posed by Daniel Bernoulli to solve St.Petersberg paradox. He quoted that *"The determination of the value of an item must not be based on the price, but rather on the utility it yields. . . . There is no doubt that a gain of one thousand ducats is more significant to the pauper than to a rich man though both gain the same amount"* [39]. This concept introduced a new term called *utility* that captures how an agent evaluates the outcome of a choice. Specifically, Bernoulli developed a discounted utility framework where the agent valuates tasks differently across time. However, such a model does not capture the temporally changing goals of an agent, potentially due to fatigue.

A few centuries later, John von Neumann and Oskar Morgenstern [6] stated nine axioms under which a rational agent with a total preference order always picks choices so as to maximize their expected utility. Even after seven decades of technical advancements, most researchers continue to model humans as utility maximizers because of mathematical tractability. However, a rational agent with such a model can search indefinitely among the available options until he/she finds the best option. Such a behavior contradicts human decision-making, especially since people often experience *decision fatigue* after a long session of decision making and eventually avoid making a decision. Such cognitive limitations of the human brain, which makes them deviate from the EUM behavior, are accounted by other behavioral models, which are classified broadly as *Boundedly Rational* models.

2.4. BOUNDED RATIONALITY AND SATISFICING

Bounded rationality assumes that people make decisions under diverse physical and cognitive limitations (e.g. problem size, limited memory). Herbert Simon [8] first coined the term to explain how humans deviate from EUM because they experience limits in formulating and solving complex problems and in processing (receiving, storing, retrieving, transmitting) information. He introduced the concept of *satisficing* to better describe the decision-making strategy of the human agent. Satisficing defines that the human agent stops searching among the options as soon as an acceptability threshold is met. The threshold value is intrinsic to the agent and depends upon his/her personality. In the past, extensive work has been done in understanding satisficing concept [8], mathematically modeling it [40] and how different it is from EUM [41, 42]. Effects of satisficing behavior in various domains have also been studied extensively [43, 44, 45], and techniques have been proposed to avoid this behavior as it leads to sub-optimal decisions [46].

A very relevant work to our thesis is done by Reverdy *et.al.,*, in [47], where they studied satisficing in multi-armed bandit problems. They investigated the concept of satisficing on two utility constructs: (i) mean reward, and (ii) instantaneous reward for choosing an option. They showed that the satisficing model defined based on each of these utility constructs is equivalent to the standard exploration/exploitation trade-off problem in multi-armed bandit problems with novel regret minimization notions corresponding to the utility constructs and presented bounds on performance. However, in the entire work, they believed that that the chosen threshold remains constant throughout the decision process.

However, in our work, we question the assumption made by the satisficing model that the threshold remains stationary over time. We believe that people experience discontent as time progresses and diminishes the threshold over time, making the agent satisfied much earlier than intended. This kind of behavior can be observed very frequently in our daily life. For instance, a driver in a ride-sharing company accepts fewer ride requests by the end of the day compared to the start of the day. To capture this diminishing behavior of human agents' threshold levels, we propose a novel heuristic called *Discounted Satisficing* and propose novel algorithms to estimate model parameters and predict agent's stopping time. This heuristic is primarily designed to predict crowd workers' stopping time that performs hundreds of micro tasks every day.

2.5. CROWDSOURCING

Crowdsourcing is an online mechanism where any registered agency can outsource tasks to a large pool of unknown workers in the form of an open call on the Internet. In such a framework, the decisions made by workers can be broadly classified into three types:

- Pick tasks that suit according to their preferences.
- Make executive decisions in completing the task.
- Decide a stopping time beyond which the agent temporarily quits from the crowdsourcing platform.

Current literature focuses on extensively analyzing the first type of decisions using various models, such as utility maximization and satisficing [46]. These studies have led to the design of many recommender systems for crowdsourcing platforms [3, 4, 5, 48, 49]. The second type of decision is analyzed by evaluating the performance quality of the workers [50]. However, there is very little work on how/when a crowd worker decides to stop working temporarily on the crowdsourcing platform. Therefore, in our work, we model agents' decisions about their stopping times using *discounted satisficing* heuristic on a sequential multi-arm bandit framework to model human factors in workers' decisions. For simplicity and tractability, we focus only on investigating stopping times by ignoring the first two types of decision processes. By predicting the stopping time, we plan to design personalized recommendation systems for crowdsourcing platforms and effective interventions to help workers work on content moderation tasks, thereby attempting to decrease the PTSD problems.

In the remaining sections, we use the crowdsourcing platform and crowd workers as an application domain to develop and validate learning algorithms to estimate parameters in our proposed discounted satisficing heuristic, and predict agent stopping time.

3. DISCOUNTING SATISFICING TO MODEL AND PREDICT CROWD WORKERS' STOPPING TIME

3.1. NOTATIONS AND DEFINITIONS

Consider a crowdsourcing $\mathcal{P}$ where a worker $\mathcal{W}$ chooses one of the tasks from a set $C = \{1, \cdots, N\}$. Assume that the worker evaluates each of these tasks based on a set of attributes $\mathcal{M} = \{1, \cdots, m\}$. In other words, if the worker $\mathcal{W}$ chooses the i^{th} task at time $k \in \mathcal{T}$, then he/she observes a multi-attribute utility

$$u_{i,k} = \sum_{j \in \mathcal{M}} \alpha_j \, x_{i,j,k}, \tag{3.1}$$

where α_j is the weight given to the j^{th} attribute in $\mathcal{M}$, and $x_{i,j,k} \in \mathbb{R}_+$ denotes the instantaneous reward (single-attribute utility) for choosing the i^{th} task at time $k \in \mathcal{T}$ with respect to the j^{th} attribute in $\mathcal{M}$. In this thesis, we ignore the rationality behind the crowd worker's decision to choose i^{th} task, since our goal is to predict his/her stopping time. Let

$$U_t = \sum_{k=1}^{t} u_{i,k} \tag{3.2}$$

denote the total accumulated utility at crowd worker $\mathcal{W}$ after t time periods.

Definition 1. An agent is said to exhibit *discounted satisficing heuristic*, if there exists two numbers $\lambda \in \mathbb{R}_+$ and $\beta \in (0, 1]$ such that the stopping time t^* is given by

$$t^* = \text{minimize} \ \left\{ t \in \mathcal{T} \mid U_t = \sum_{k=1}^{t} u_{i,k} \geq \beta^{t-1} \lambda \right\}, \tag{3.3}$$

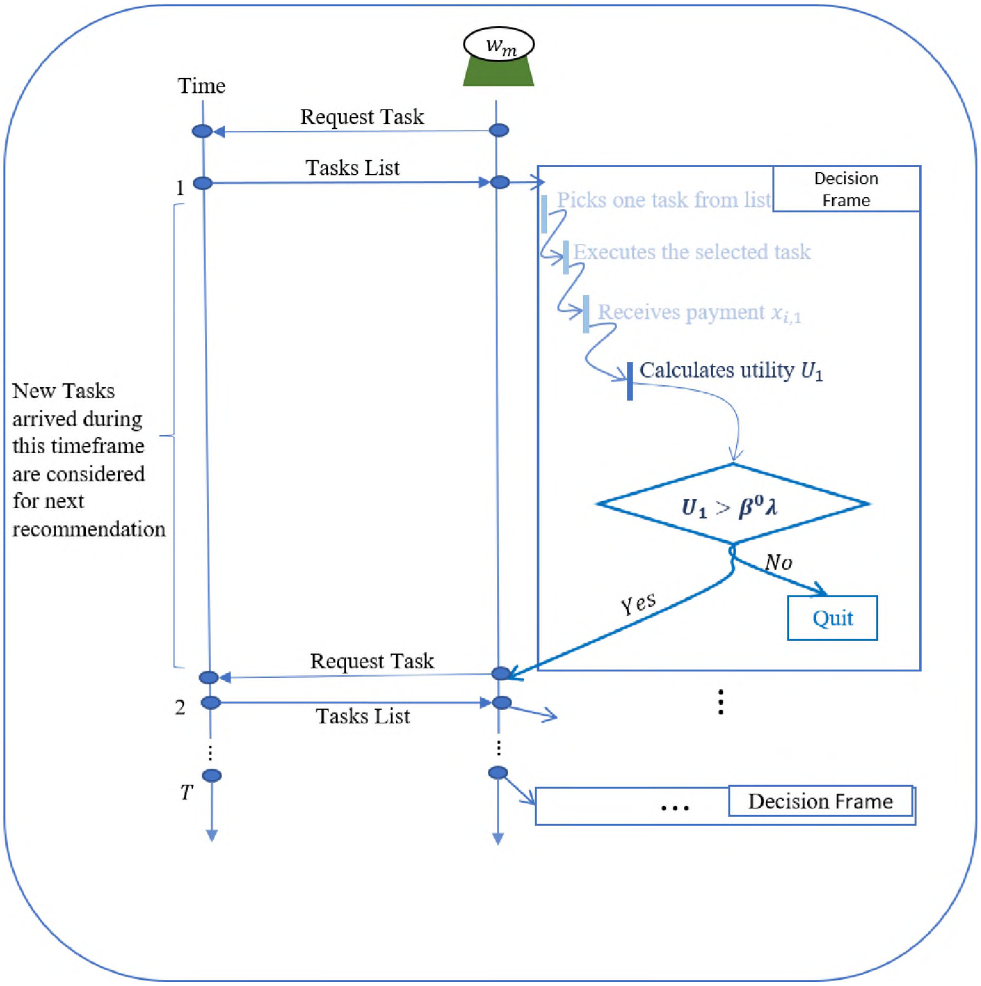

Figure 3.1. Sequential decision-making strategy of a crowdworker exhibiting Discounted Satisficing Heuristic

In the above definition 1, the parameter λ represents the total utility desired by the worker $\mathcal{W}$ before the commencement of their work-day on platform $\mathcal{P}$. On the other hand, the parameter β captures the worker's discounting behavior over time due to the increasing weariness levels. Figure 3.1 shows the flowchart of the sequential decision-making strategy employed by a crowd worker using discounted satisficing heuristic at a discrete-time unit. At every decision time, the worker is presented with a set of recommended tasks by the crowdsourcing platform, which are then carefully examined before picking a choice. The worker executes the task depending on his cognitive capacity and submits the task to the task producer and receives payment for successful submission. He/She then decides whether to continue working on the platform or quit by comparing the accumulated utility with the discounted threshold. It is evident from discounted satisficing heuristic that the stopping time of the crowd worker depends on his/her intrinsic parameters like threshold/discontent levels/weights they assign to each attribute (λ, β, α). However, in reality, getting satisfied is an involuntary process, and humans are unaware of the parameters' exact values. Hence, the model parameters need to be estimated before predicting the stopping time.

In this thesis, we develop algorithms that first estimates the values of the model parameters and use the estimated values to predict the stopping time of the crowd worker using a discounted satisficing heuristic.

3.2. LEARNING ALGORITHMS TO PREDICT STOPPING TIME

From Definition 1, the stopping time is associated with a system of multivariate non-linear inequalities based on the sequence of decisions employed until stopping time t^*. So, estimating all the parameters in one step, and predicting the stopping time is not easy. Hence, we started our study with a trivial case that assumed the utility function of the crowd worker as a single attributed function and later extended it to multi-attribute with known weights and unknown weights. We developed algorithms to predict the crowd worker's stopping time $\mathcal{W}$, and below subsections cover the in-depth details.

3.2.1. Single Attribute Utility. [1]In this setting, we attempt to predict the stopping time of the crowdworker($\mathcal{W}$) in a simple scenario, where we assumed that he/she evaluates the tasks based on only one attribute ($m = 1; \alpha_1 = 1$) and hence the utility observed by him/her for choosing i^{th} task at time $k \in \mathcal{T}$ is equal to the instantaneous reward obtained for the single attribute ($u_{i,k} = x_{i,k}$). In such a scenario, the stopping time of the crowdworker using discounted satisficing heuristic is defined as below:

[1]This work has been presented as a work-in-progress poster at HCOMP'2019 [51]

$$t^* = \text{minimize} \left\{ t \in \mathcal{T} \mid U_t = \sum_{i=1}^{t} x_{i,t} \geq \beta^{t-1}\lambda \right\} \tag{3.4}$$

where $x_{i,t}$ is the immediate reward observed by the crowd worker $\mathcal{W}$ for executing i^{th} task at time $k \in \mathcal{T}$ with respect to the single attribute that he/she takes into consideration while evaluating the task. Assuming the utilities of the tasks are perfectly observable, a given stopping time t^* is associated with a sequence of non-linear inequalities based on the sequence of stopping decisions made by the worker according to Definition 1.

$$\begin{aligned} &\beta^{i-1}\lambda > U_i, \ \forall\, i = 1, \cdots, t^* - 1, \\ &\text{and } \ \beta^{t^*-1}\lambda \leq U_{t^*} \end{aligned} \tag{3.5}$$

The above system of non-linear inequalities can be linearized by applying logarithms on both sides to obtain a polytope described by the following system of linear inequalities:

$$\begin{aligned} &(i-1)\log\beta + \log\lambda > \log U_i, \ \forall\, i = 1, \cdots, t^* - 1, \\ &\text{and } \ (t^* - 1)\log\beta + \log\lambda \leq \log U_{t^*} \end{aligned} \tag{3.6}$$

A natural way to compute the parameter estimates $(\hat{\lambda}, \hat{\beta})$ is to consider the centroid of the above polytope as a candidate solution. However, the above polytope is not necessarily compact, making it impossible to employ this method in general. Therefore, we assume the polytope to be compact via imposing limits on β, i.e.

$$0 < \beta_L \leq \beta \leq \beta_U < 1, \tag{3.7}$$

where β_L, β_U can be justified as prior knowledge about the worker $\mathcal{W}$. This estimate can be further improved via observing worker's decisions over multiple iterations[2]. Figure 3.2 depicts the pictorial representation of the convex/compact polytope formed by the system of linear inequalities associated with the stopping time(t^*).

[2]For example; each iteration could represent a sequence of decisions made by the worker over a single day.

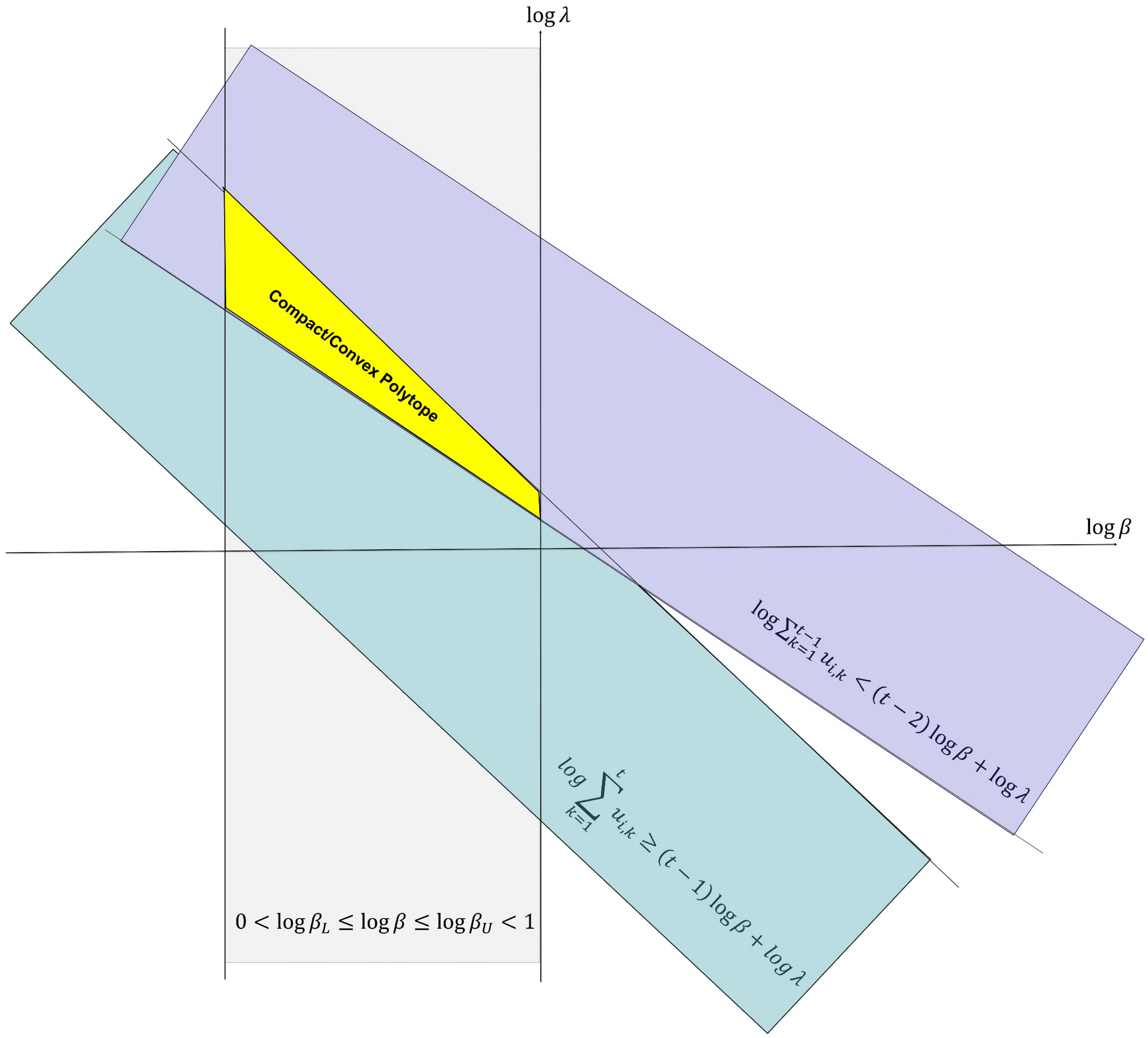

Figure 3.2. Convex/Compact Solution Polytope

Let $D_n = (d_1, d_2, \cdots, d_n)$ denote the worker's decisions over n iterations, where $d_j = \{u_{i,1},\ u_{i,2}, \cdots,\ u_{i,T_j}\}$ contains the sequence of utilities obtained by worker $\mathcal{W}$ until he/she stops at time T_j in the j^{th} iteration, for any $j = 1, \cdots, n$. Since each data tuple d_j produces a compact polytope (denoted as R_j) from Equations (3.6) and (3.7), we obtain a reduced polytope $R = R_1 \cap \cdots \cap R_n$ from the intersection of the polytopes obtained from n iterations, which can be efficiently computed using Sutherland-Hodgman Algorithm [52].

We evaluated our proposed algorithm on simulation data (details presented in Section 4), where we observed that the algorithm performs better when $\beta > 0.5$ and unable to predict stopping time accurately in case of lower β values ($\beta < 0.5$). This can be attributed to the fact that the dynamic thresholds of the workers with lower β values generally deteriorate at a much faster rate, thereby revealing very little about the model parameters in their choices. Besides, we have noticed deteriorated performance in the presence of small values of λ, which can be justified with similar reasons, as stated above, in the case of small values of β. Hence, we need an algorithm to accurately predict the stopping time, even in the presence of little information about the worker. Similarly, results on real data showed only 26% accuracy when the utilities defined as immediate rewards obtained by the workers (more details presented in Section 4). However, in practice, people's utility functions are known to constitute preferences across multiple attributes. Hence, in the remaining sections, we assumed that the worker's utility function is a multi-attribute function and developed algorithms that can accurately predict stopping time even in the presence of less information about the worker.

3.2.2. Multi Attribute Utility with Known Weights. As discussed in the previous section, people evaluate the alternatives available to him/her over multiple attributes and then make a decision. For instance, people usually consider price, quality, quantity, and product reviews before deciding to buy a product on an e-commerce platform. This evaluation to make a decision becomes more complicated with the increase in the number of attributes taken into consideration while evaluating a choice and draining out more energy from the human. If the person has to make a sequence of such decisions, then he/she will experience decision fatigue and avoids deciding. Hence, predicting the human agent's stopping time in the presence of a large number of attributes can help avoid the decision avoidance stage. Hence, in this section, we try to model the utility function of the crowdworker as a multi-attribute function and develop algorithms to predict his/her stopping time.

Mathematically, let the utility of the crowd worker to be a linear combination of single-attribute utilities as shown below.

$$U_{i,t} = \sum_{i=1}^{t} \sum_{k \in \mathcal{M}} \alpha_k x_{i,k} \geq \beta^{t-1}\lambda, \tag{3.8}$$

where the weights $\{\alpha_k\}_{k \in \mathcal{M}}$ are known. Assuming that the workers preferences over M task attributes and the rewards associated with each of them are perfectly observable, the stopping time t^* can be modelled as

$$t^* = \text{minimize } \left\{ t \in \mathcal{T} \;\middle|\; U_{i,t} \geq \beta^{t-1}\lambda \text{ and } \alpha \text{ is known} \right\}, \tag{3.9}$$

As the weights allocated to each attribute and the immediate reward associated with them are perfectly observable, the above modeling of stopping time leads to a system of bi-variate(λ, β) non-linear inequalities associated with each decision until stopping time as mentioned in Equation 3.10.

$$\begin{aligned} U_{i,t} &< \beta^{t-1}\lambda, \qquad i \in \{1, \cdots, t^* - 1\} \\ \text{and } U_{1,t^*} &< \beta^{t^*-1}\lambda \end{aligned} \tag{3.10}$$

We propose below three algorithms to predict the stopping time in case of such a multi-attribute utility function along with estimating the model parameters λ, β.

- Online Learning Based on Bounds (OL-BB)
- Batch Learning using Quadratic Programming (BL-QP)
- Online Learning using Quadratic Programming (OL-QP)

3.2.2.1. Online learning based on bounds (OL-BB). The algorithm leverages the fact that every decision employed by worker $\mathcal{W}$ until stopping time t^* on a particular iteration/day establish either a lower/upper bound on the discounted threshold $\beta^p\lambda, p \in$

$\{1, \cdots, t^* - 1\}$ in terms of the utility observed. Using the two decision times with most tightly bounded discounted thresholds, it estimates the model parameters and consequently predicts the stopping time.

More formally explaining the algorithm, let $d_j = \{U_{i,1}, \cdots U_{i,t_j}\}$ denote the worker's utilities for the decisions employed over j^{th} iteration, where $U_{i,t} = (x_{i,1,t}, \cdots x_{i,\mathcal{M},t})$ sequence of instantaneous rewards observed by the worker $\mathcal{W}$ with respect to $\mathcal{M}$ attributes for executing i^{th} task. The system of bi-variate non-linear inequalities associated with each decision until stopping time (t_j^*) establishes a range to the discounted threshold as shown in the Equation (3.11). Let $(Max, Min) \in \mathbb{R}_+$ be any two integers, then

$$\begin{aligned} & U_{i,t} < \beta^{t-1}\lambda < Max, \quad t \in \{1, \cdots, t_j^* - 1\} \\ \text{and} \quad & Min < \beta^{t_j^*-1}\lambda \geq U_{i,t_j^*} \end{aligned} \tag{3.11}$$

Let e, f be the the decision times where we observe tighter bounds on the discounted threshold using the system of inequations as specified in the Equation (3.11). Using the bounds at the the decision times e, f, the model parameter values $(\hat{\lambda}, \hat{\beta})$ are estimated as below:

$$\begin{aligned} \hat{\beta} &= \left(\frac{mean(U_{i,e}, Max)}{mean(U_{i,f}, Min)}\right)^{e-f} \\ \hat{\lambda} &= \frac{U_{i,e}}{\beta^e} \end{aligned} \tag{3.12}$$

The prediction can be poor during the initial iterations, but can be improved by observing workers' decisions over multiple iterations. Let $Bounds = \{i : (a_i, b_i) \mid \forall i \in (0, max\{t_1^*, \cdots, t_j^*\})\}$ be the set of bounds at each decision time over j iterations. When a new iteration is made (d_{j+1}), the bounds on discounted threshold are updated, as below:

$$
\begin{array}{rcll}
U_{i,t} & < \beta^{t-1}\lambda < & Max & \text{, if } t \in \{1,\cdots,t^*_{j+1}-1\} \text{ and } t \notin Bounds \\
max\{a_t, U_{i,t}\} & < \beta^{t-1}\lambda < & Max & \text{, if } t \in \{1,\cdots,t^*_{j+1}-1\} \text{ and } t \in Bounds \\
Min & < \beta^{t^*_{j+1}}\lambda < & U_{i,t^*_{j+1}} & \text{, if } t^*_{j+1} \notin Bounds \\
Min & < \beta^{t^*_{j+1}}\lambda < & min\{b_{t^*_{j+1}}, U_{i,t^*_{j+1}}\} & \text{, if } t^*_{j+1} \in Bounds
\end{array}
\tag{3.13}
$$

With each iteration, this online learning algorithm updates bounds on $\beta^p\lambda, p \in \{0,\cdots,max\{t^*_1,\cdots,t^*_j\}\}$, where j being the current iteration using the above Equation (3.13) and uses the final calculated bounds to estimate model parameters. Algorithm 1 describes the proposed online algorithm to predict the stopping time by estimating the model parameters.

Using this algorithm, we were able to predict the stopping time accurately in the case of higher and lower β values on the simulated data (details in Section 4). However, in the presence of lower β values, the algorithm fails to accurately estimate the β parameter value. We propose another algorithm to predict the stopping time in the presence of multi-attribute utility function with known weights.

3.2.2.2. Batch learning using quadratic programming (BL-QP). As discussed in the previous section, the stopping time (t^*_j) is associated with a system of bivariate non-linear inequations that can be transformed into a system of bivariate linear inequations using log-linearization. Now that we have a linear system, we formulated the prediction problem as a quadratic program [53] to minimize the gap between the discounted threshold of the worker and the accumulated utility observed at each decision time until stopping time. The optimal solution of the minimization problem is considered as the estimated values of model parameters ($\hat{\lambda}, \hat{\beta}$). While estimating the model parameters, this algorithm ingests all the training data available about the worker at a time to estimate the parameter values

Algorithm 1 Online learning based on bounds (OL-BB)

procedure OL-BB(d_{j+1}) ▷ d_{j+1} being current iterations' data
  Required: $Max \leftarrow Very\ Large\ Integer$
  Required: $Min \leftarrow Very\ Small\ Integer$
  Required: $Bounds \leftarrow \{i : (a_i, b_i) \mid \forall i \in (0, max(t_1^*, \cdots, t_j^*)\}$
  ▷ (a_i, b_i) be the lower&upper bounds of $\beta^i \lambda$ at each decision time until the current iteration
  procedure OL-BB(d_{j+1})
    for every $k \in d_{j+1}$ **do**
      if $k == len(d_{j+1})$ **then**
        if $k \in Keys(Bounds)$ **then**
          $b_k \leftarrow min(b_k, U_{i,k})$
        else
          $Bounds[k] \leftarrow (Min, U_{i,k})$
      else
        if $k \in Keys(Bounds)$ **then**
          $a_k \leftarrow max(a_k, U_{i,k})$
        else
          $Bounds[k] \leftarrow (U_{i,k}, Max)$
  $diff \leftarrow \{i : b_i - a_i \mid \forall i \in Keys(Bounds)\}$
  $avg \leftarrow \{i : (b_i + a_i)/2 \mid \forall i \in Keys(Bounds)\}$
  $\hat{\beta} \leftarrow \left(\frac{avg[e]}{avg[f]}\right)^{e-f}$ ▷ Assuming e, f are the decision times with tighter bounds
  $\hat{\lambda} \leftarrow \frac{U_e}{\beta^e}$
  return $\hat{\lambda}, \hat{\beta}$

and hence called batch learning algorithm. Using estimated values and the stopping time Equation (3.9), the stopping time $\hat{t}^*$ is predicted. Problem (P1) shows the minimization problem that is used to predict the stopping time.

Let $\boldsymbol{x} = [\log\lambda \quad \log\beta \quad s_{1,1} \quad \cdots \quad s_{1,(t_1)^*} \quad s_{2,1} \quad \cdots \quad s_{n,(t_n)^*}]^T$ denote the vector of unknown model parameters where $s_{i,j}$ represents slack variables used to convert the inequality associated with j^{th} decision on i^{th} iteration into an equation. The slack variables also capture any variations in the assumed utility function. If we define

$$A = \begin{bmatrix} 1 & 0 & -1 & 0 & \cdots & \cdots & \cdots & 0 \\ 1 & 1 & 0 & -1 & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 \\ 1 & t_1^* - 1 & 0 & \cdots & 1 & \cdots & \cdots & 0 \\ 1 & 0 & 0 & 0 & \cdots & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_n^* - 1 & 0 & 0 & \cdots & \cdots & \cdots & 1 \end{bmatrix}, \quad \text{and} \quad b = \begin{bmatrix} \log U_{1,1} \\ \log U_{1,2} \\ \vdots \\ \log U_{1,t_1^*} \\ \log U_{2,1} \\ \vdots \\ \log U_{n,t_n^*} \end{bmatrix}$$

then the goal is to minimize the distance between discounted threshold values and utility obtained at each decision over multiple iterations

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \|A\boldsymbol{x} - b\|_2^2 \\ \text{subject to} \quad & 1.\ 0 < [1, 0, \mathbf{0}_{t_1^*+\cdots+t_n^*}]^T \boldsymbol{x} < \infty \\ & 2.\ -\infty < [0, 1, \mathbf{0}_{t_1^*+\cdots+t_n^*}]^T \boldsymbol{x} \leq 0 \\ & 3.\ \{s_{i,j} \geq 0 \mid \forall\, i \in \{1, \cdots, n\} \text{ and } j \in \{t_1^*, \cdots, t_n^*\}\} \end{aligned} \tag{P1}$$

The constraints impose limits on model parameters and the slack variables to restrict the acceptable solution space to positive quadrant(refer Definition 1).

$$\begin{aligned} \lambda > 0 \quad & \implies 0 \leq \log \lambda < \infty \\ \beta \in (0, 1) \quad & \implies -\infty < \log \beta \leq 0 \end{aligned} \tag{3.14}$$

The above formed problem (P1) is a standard quadratic program which can be solved using interior-point algorithms [53]. Interior-point algorithms converts the original minimization problem with linear inequality constraints into an unconstrained optimization problem using a barrier function that includes inequality constraints in the objective function as a penalizing term. Barrier function is a continuous function whose value on a point increases to infinity as the point approaches the boundary of the feasible region of an optimization problem. The two most common types of barrier functions are inverse barrier functions and logarithmic barrier functions. Now that the problem becomes an unconstrained optimization problem and techniques like gradient descent can be used to compute the optimal solution. The near optimal solution $\hat{\boldsymbol{x}} = [\log \hat{\lambda}, \log \hat{\beta}, \hat{s_{1,1}}, \cdots, \hat{s_{n,t_n^*}}]^T$ for Problem (P1) is calculated using interior-point algorithms as mentioned earlier. The solution describes the estimated values of model parameters and using the estimated values $(\hat{\lambda}, \hat{\beta})$ and the stopping time Equation (3.9), the stopping time $\hat{t^*}$ is predicted.

Simulation results (as explained in Section 4) show that the model estimates stopping time accurately in both the cases of lower and higher β values. However, similar to the previous algorithm, this algorithm fails to estimate the β parameter values accurately when $\beta < 0.5$.

Another disadvantage using this algorithm is that it is a batch learning algorithm ie., estimates the model parameters' values using the entire training data at once. However, in reality, the worker's decision data over multiple iterations are available in sequential order and the proposed algorithm cannot accommodate the patterns in the new iterations' data on-the-fly. It has to be retrained on the entire data (old+new) to accommodate new patterns. This thought has made us design below online learning algorithm to predict the stopping time in the presence of multi-attribute utility value with known weights.

3.2.2.3. Online learning using quadratic programming (OL-QP). The previously proposed batch learning algorithm assumes that the workers' decisions over multiple iterations are available before train the model. However, in the real world, data might be observed on-the-fly, and the model should be able to accommodate the new data in its' structure. Online algorithms are the most commonly used techniques in machine learning in such scenarios. Hence we propose an online learning algorithm that ingests workers' decision data of only one iteration at a time and predicts the stopping time and estimating the model parameters.

This online learning algorithm uses regularized convex programming technique to predict the stopping time whose objective is to minimize the regularized regret that measures the distance between the utilities observed at the worker and the discounted threshold in the current iteration. To apply this technique, first the non-linear system of inequalities associated with each decision on a particular iteration is converted into linear equations by log-linearizing and adding slack variables $\{s_{i,1}, \cdots s_{i,t_i^*}\}$, where i is the current iteration.

Let $\boldsymbol{x_i} = [\log \lambda, \log \beta, s_{i,1}, \cdots, s_{i,t_i^*}]$ be the vector representing the model parameters in the i^{th} iteration. If we define

$$A_i = \begin{bmatrix} 1 & 0 & -1 & \cdots & \cdots & 0 \\ 1 & 1 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_i^* - 1 & 0 & \cdots & \cdots & 1 \end{bmatrix}, \quad b_i = \begin{bmatrix} \log U_{i,1} \\ \log U_{i,2} \\ \vdots \\ \log U_{i,t_i^*} \end{bmatrix}, \text{ and}$$

$$I_i = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{t_i^* \times t_i^*}$$

and let R_i be the regret observed while estimating the parameters in the i^{th} iteration,

$$R_i = \|A_i \boldsymbol{x_i} - b_i\|_2^2 \tag{3.15}$$

then the goal of the algorithm is to minimize the regret observed over the iteration,

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & R_i + \eta \left\|\boldsymbol{x_i} - \boldsymbol{x_{i-1}}\right\|_2^2 \\ \text{subject to} \quad & 1.\ 0 < [1, 0, \boldsymbol{0}_{t_i^*}]^T \boldsymbol{x_i} < \infty \\ & 2.\ [0, 1, \boldsymbol{0}_{t_i^*}]^T \boldsymbol{x_i} \leq 0 \\ & 3.\ [0, 0, I_i]^T \boldsymbol{x_i} \geq \boldsymbol{0}_{t_i^*}{}^T \end{aligned} \tag{P2}$$

where t_i^* is the stopping time of the current iteration, and η is the tuning parameter for regularization to penalize the model for overfitting. The above formed problem (P2) is a standard quadratic program which can be solved using interior-point algorithms [53] which is very similar to the procedure described while presenting batch learning algorithm proposed earlier. The near optimal solution of this problem is considered as the values of model parameters in the current iteration and the estimated values are updated as a weighted sum of model parameter values from the previous iteration and current iteration. Algorithm 2 describes the step-by-step procedure.

Algorithm 2 Online Learning using Qudratic Programming (OL-QP)

1: **procedure** OL-QP(d_{j+1}) ▷ $d_{j+1} \leftarrow$ set of utilities in current iteration
2: **Required:** $\hat{\lambda}_i$ and $\hat{\beta}_i$ ▷ Estimates from previous iteration
3: $\lambda, \beta \leftarrow$ Solve Problem (P2) using d_i ▷ Estimate current parameters
4: $\hat{\lambda}_{i+1} \leftarrow \theta * \hat{\lambda}_i + (1-\theta) * \lambda,$ ▷ Update rule for model parameter λ
5: $\hat{\beta}_{i+1} \leftarrow \theta * \hat{\beta}_i + (1-\theta) * \beta,$ ▷ Update rule for model parameter β
6: **return** $\hat{\lambda_{i+1}}, \hat{\beta_{i+1}}$

Results show that the proposed algorithm has a similar performance in predicting the stopping time to that of the other two algorithms that we described earlier in this section(OL-BB, BL-QP). It estimates the stopping time accurately; however fails to estimate both the λ, β parameters in case of lesser $\beta < 0.5$ values(detailed analysis presented in Section 4).

3.2.3. Multi-Attribute Utility with Unknown Weights. Previously proposed algorithms in the thesis are based on the assumption that the weights allocated to each attribute taken to consideration while evaluating a task are known. In contradiction, workers generally have a preference order over the attributes. They are unaware of the exact weights that they allocate to each attribute, and hence this information is not readily available for the algorithm to proceed with prediction. When the weights associated with each attribute is unknown, the stopping time t^* is modeled, as shown in Equation (3.16).

$$t^* = \text{minimize}\,\{t \in \mathcal{T} \mid U_t = \sum_{i=1}^{t} \sum_{k \in \mathcal{M}} \alpha_k x_{i,k} \geq \beta^{i-1}\lambda\} \tag{3.16}$$

From the above equation, it is clear that the stopping time is associated with a system of non-linear multivariate inequations corresponding to each decision until stopping time.

$$\begin{aligned} &\sum_{i=1}^{t} \sum_{k \in \mathcal{M}} \alpha_k x_{i,k} < \beta^{t-1}\lambda, \qquad t \in \{1, \cdots t^* - 1\} \\ \text{and} &\sum_{i=1}^{t^*} \sum_{k \in \mathcal{M}} \alpha_k x_{i,k} \geq \beta^{t^*-1}\lambda \end{aligned} \tag{3.17}$$

The problem of estimating the model parameters in such a non-linear multivariate environment can quickly become an ill-posed problem if the number of attributes is higher than the number of tasks the worker executes in an iteration, thereby making the designing of the algorithm more difficult. Hence we propose a batch learning algorithm that uses alternating minimization techniques(BL-AM) to estimate the model parameters.

From Equation (3.16), the stopping time is dependent on multiple variables($\lambda, \beta, \boldsymbol{\alpha}$) that need to be estimated before predicting the stopping time. The problem of estimating model parameters forms a quadratic program where our goal is to minimize the root-mean-squared-error of the predicted stopping time observed over the entire training data.

$$\underset{\hat{\lambda},\hat{\beta},\hat{\alpha}}{\arg\min} \quad A(\hat{\lambda}, \hat{\beta}, \hat{\alpha}) = \sqrt{\mathbb{E}\Big(\sum_{k=1}^{n} (t_k^* - \hat{t}_k^*)^2 \Big)} \tag{P3}$$

Optimizing over multiple variables jointly makes the problem intractable and difficult to compute. One way to handle this optimization problem is to combine all the unknowns into a single variable $\boldsymbol{x} = (\alpha, \beta, \boldsymbol{\alpha})$ and directly applying standard iterative algorithms like gradient descent. An alternative approach to solving such problems is to adapt *alternating minimization* technique, which sequentially optimizes one variable while keeping the others constant. Compared to standard algorithms, alternating minimization algorithms are easy to implement because subproblems are easy to handle, can give a closed-form solution, and has a better convergence rate. Hence, we use this technique to design an algorithm to predict the stopping time. The algorithm we propose formulates the main optimization problem (P3) as a series of two subproblems. The first subproblem assumes that the weights of the attributes are known and estimates the threshold and discount factor values. The second problem keeps the threshold and discount factor as constants and estimates the weights of the attributes using interior-point algorithms. This alternating procedure of estimating parameter values is repeated multiple times until a convergence in the root-mean-squared error of the predicted stopping time is observed. Summarizing the proposed algorithm as: starting at the arbitrary initial point where all the attributes had equal weightage($\boldsymbol{\alpha_0}$); for $k \geq 1$, iteratively compute

$$\begin{aligned} \hat{\lambda}_k, \hat{\beta}_k &\in \operatorname*{arg\,min}_{\lambda,\beta \in \mathbb{R}_+} A(\lambda, \beta, \hat{\alpha}_{k-1}) \\ \hat{\alpha}_k &\in \operatorname*{arg\,min}_{\alpha \in R^m_+} A(\lambda_k, \beta_k, \alpha) \end{aligned} \tag{P4}$$

The two minimization problems specified in the Problem (P4) are solved using solutions approaches available to solve quadratic programs. The first subproblem is equivalent to the problem we formulated in the Section 3.2.2, where we assumed the utility of the crowdworker as a multi-attribute utility with known weights. We use one of the algorithms that we proposed earlier(Section 3.2.2.2) to solve this subproblem. The second subproblem, where values of β, λ are assumed to be known, the stopping time(t^*) is associated with a system of linear inequalities with the weights as unknowns. The linear inequalities, when transformed into linear equations using slack variables, the optimization problem to learn the weights associated with each attribute, is a standard quadratic program. An near approximate solution is calculated using interior-point algorithms [53]. The quadratic program is mathematically formulated as shown in Problem P5.

Let $\boldsymbol{x} = [\alpha_1, \cdots, \alpha_m, s_{1,1}, \cdots, s_{n,t_n^*}]$ be the vector representing the model parameters(weights) and the slack variables added to transform the linear inequation to linear equations. If we define

$$A = \begin{bmatrix} x_{1,1,1} & \cdots & x_{1,1,\mathcal{M}} & 1 & \cdots & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^{t_1^*} x_{1,i,1} & \cdots & \sum_{i=1}^{t_1^*} x_{1,i,\mathcal{M}} & 0 & \cdots & -1 & \cdots & \cdots & 0 \\ x_{2,1,1} & \cdots & x_{2,1,\mathcal{M}} & 0 & \cdots & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^{t_n^*} x_{n,i,1} & \cdots & \sum_{i=1}^{t_n^*} x_{n,i,\mathcal{M}} & 0 & \cdots & \cdots & \cdots & \cdots & -1 \end{bmatrix}, \quad b = \begin{bmatrix} \lambda \\ \beta\lambda \\ \vdots \\ \beta^{t_1^*-1}\lambda \\ \lambda \\ \vdots \\ \beta^{t_n^*-1}\lambda \end{bmatrix}$$

our goal is to minimize the distance between the observed utilities and the discounted threshold,

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \|A\boldsymbol{x} - b\|_2^2 \\ \text{s.t} \quad & 1.\ [\mathbf{1}_{\mathcal{M}}, \mathbf{0}_{t_1^*+\cdots+t_n^*}]\boldsymbol{x} = 1 \\ & 2.\ \boldsymbol{x} \geq 0 \end{aligned} \tag{P5}$$

The constraints restrict the solution space to positive quadrant since weight associated with every attribute and the slack variables are always positive and sum of the all the weights is equal to one. The above problem (P5) is in a standard quadratic problem form which can be solved using interior-point algorithms [53]. The near optimal solution of this problem calculated using interior-point algorithm using barrier function technique is considered as the estimated values of the model parameters. Using the estimated values and discounted satisficing heuristic, stopping time(t^*) is predicted. Algorithm 3 gives a step-by-step procedure of the proposed alternating minimization algorithm.

Algorithm 3 Batch Learning using Alternating Minimization(BL-AM)

procedure BL-AM($\mathcal{D}$)
  Required: Training Data $\mathcal{D} = [d_1, \cdots, d_n]$
  Initialize $\hat{\alpha} = \left[\frac{1}{||\mathcal{M}||}, \cdots, \frac{1}{||\mathcal{M}||}\right]$ ▷ $\mathcal{M}$ being the set of attributes
  while Convergence on regret(R) is not seen **do**
    $\hat{\lambda}, \hat{\beta} \leftarrow$ Solve Problem (P1) using $\mathcal{D}, \hat{\alpha}$
    $\hat{\alpha} \leftarrow$ Solve Problem (P5) using $\mathcal{D}, \hat{\lambda}, \hat{\beta}$
    for every $d_j \in \mathcal{D}$ **do**
      $\hat{t}_j^* \leftarrow$ Calculate stopping time using Equation (3.16) for d_j
    $R \leftarrow$ Calculate root-mean-squared error of regret
  return $\hat{\lambda}, \hat{\beta}, \hat{\alpha}$

Simulation results (as discussed in Section 4) reveal that the proposed algorithm could predict the values of stopping time accurately in case of both lower and higher β values. However, it fails to estimate the values of β parameter accurately in case of lower $\beta < 0.5$ values due to the presence of little information about the worker. This behavior is consistent in all the algorithms that we proposed in this thesis. Nevertheless, the predicted stopping time converges to the true value inspite of the fact that the model parameter($beta$) converge to biased estimate. This raised the question, "Is there a unique discounted satisficing model that admits a stopping time?". In the following theorem, we analyze this question and prove that two unique models(λ, β, α) can result in same stopping time decision, and hence there is no unique discounted satisficing model.

Theorem 1. There is no unique discounted satisficing model (λ, β, α) that admits the same stopping time t^* exhibited by a given crowd-worker $\mathcal{W}$ when presented with a fixed set of task rewards X.

Proof. To prove that the theorem's claim is true, consider the following two discounted satisficing models:

$$\text{Model 1:} \quad \lambda_1 = 1000; \quad \beta_1 = 0.1; \quad \alpha_{1,1} = 0.5; \quad \alpha_{1,2} = 0.5$$

$$\text{Model 2:} \quad \lambda_2 = 10; \quad \beta_2 = 0.9; \quad \alpha_{2,1} = 0.3; \quad \alpha_{2,2} = 0.7$$

Model Parameters	Decision Time								
	$t = 1$			$t = 2$			$t = 3$		
	Accumulated Utility	Discounted Threshold	Decision	Accumulated Utility	Discounted Threshold	Decision	Accumulated Utility	Discounted Threshold	Decision
$\lambda = 1000$ $\beta = 0.1$ $\alpha_1 = 0.5$ $\alpha_2 = 0.5$	$U_1 = 0.5 * 2 + 0.5 * 4 = 3$	$\beta^0\lambda = 1000$	Not Satisfied	$U_2 = 3 + (0.5 * 4 + 0.5 * 6) = 8$	$\beta\lambda = 100$	Not Satisfied	$U_3 = 8 + (0.5 * 6 + 0.5 * 8) = 15$	$\beta^2\lambda = 10$	Satisfied
$\lambda = 10$ $\beta = 0.9$ $\alpha_1 = 0.3$ $\alpha_2 = 0.7$	$V_1 = 0.3 * 2 + 0.7 * 4 = 2.88$	$\beta^0\lambda = 10$	Not Satisfied	$V_2 = 2.88 + (0.3 * 4 + 0.7 * 6) = 8.28$	$\beta\lambda = 9$	Not Satisfied	$V_3 = 8.28 + (0.3 * 6 + 0.7 * 8) = 15.68$	$\beta^2\lambda = 8.1$	Satisfied

Figure 3.3. Counter-Example: Decisions employed by the Agent

In other words, the agent following Models 1 and 2 experience utilities

$$\begin{aligned} U_t &= \alpha_{1,1}x_{1,t} + \alpha_{1,2}x_{2,t}, \text{ and} \\ V_t &= \alpha_{2,1}x_{1,t} + \alpha_{2,2}x_{2,t}, \end{aligned} \tag{3.18}$$

respectively, where $(x_{1,t}, x_{2,t})$ represents attribute-wise rewards for a given task performed at time t.

Figure 3.3 shows the decisions computed based on the above two discounted satisficing models in a choice experiment with fixed reward values

$$X = [[2, 4], [4, 6], [6, 8], [8, 10], [10, 12], \cdots].$$

The above two pairs of model parameters result in the same stopping time ($t^* = 3$). □

In other words, even though our proposed algorithms fail to estimate the model parameters values accurately in the presence of little information about the worker, they can be used to predict their stopping time decisions accurately.

4. VALIDATION

In this Section, we will see the error performance of the algorithms presented in Section 3 on simulation data and real data. We describe the simulation environment setup, real experiments conducted to collect the data and also present the web platform developed to collect real data in the case of multiple attributes.

4.1. SINGLE ATTRIBUTE UTILITY

In this subsection, as described in Section 3.2.1, we assume the worker's utility as a single attribute value. Below we present the experimental setup, data collection, and error performance of the proposed algorithm on simulation data and real data.

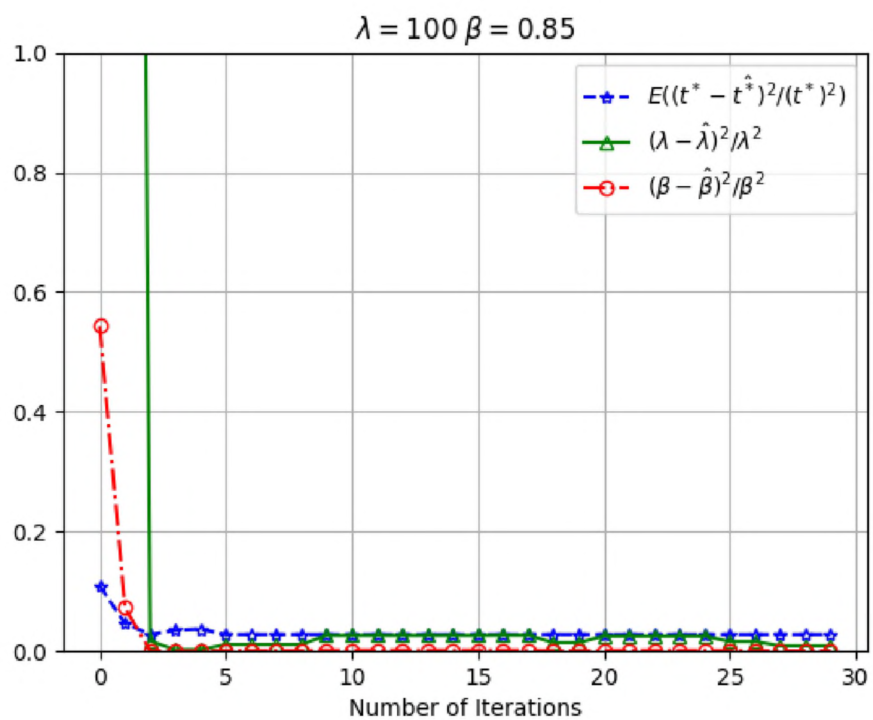

Figure 4.1. Error performance in predicting stopping time $\hat{t^*}$ and estimating $\hat{\lambda}, \hat{\beta}$ when $\lambda = 100, \beta = 0.85$

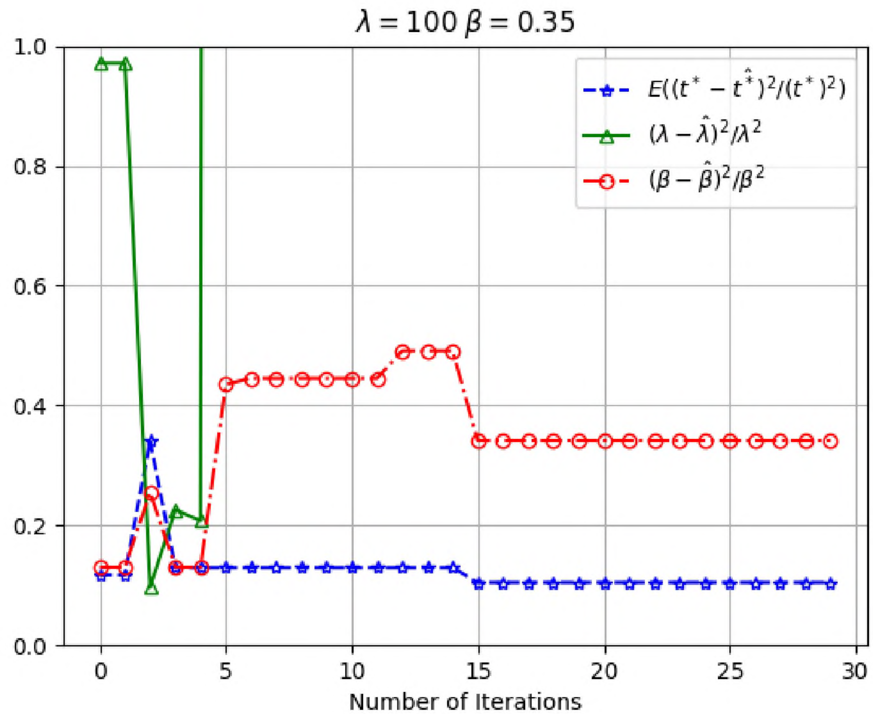

Figure 4.2. Error performance in predicting stopping time $\hat{t^*}$ and estimating $\hat{\lambda}, \hat{\beta}$ when $\lambda = 100, \beta = 0.35$

4.1.1. Validation on Simulation Data. In our simulation experiment, we assume that four tasks (arms) are available at the worker with equal probability, where the k^{th} task produces a γ-distributed reward with the shape parameter ($\alpha = 2$) and scale parameter ($\beta = 4$). Letting $\beta_L = 0.05$ and $\beta_U = 1$ in our first proposed algorithm, we run several Monte-Carlo simulations of the experiment to compute the normalized average error of the predicted stopping time and estimated model parameters. In the simulation results, we found that the estimation error of our algorithm converges to zero consistently only when $\beta \geq 0.5$. We illustrate this observation graphically using two examples both with a fixed $\lambda = 100$, where the Figure 4.1 demonstrates the convergence of estimation error to zero when $\beta = 0.85$, while the Figure 4.2 demonstrates the fact that estimation error does not converge to zero when $\beta = 0.35$. This can be attributed to the fact that the dynamic thresholds of the agents with lower β values generally deteriorate at a much faster rate, thereby revealing very little about the model parameters in their choices. Also, we have noticed deteriorated performance in the presence of small values of λ, which can also be justified with similar reasons as stated above, in the case of small values of β. However, both the Figures 4.1,4.2 show that the error while predicting stopping time converges to zero in both the cases.

4.1.2. Validation on Real Data. To validate our algorithm on real-data, we have designed an ANDROID application, where an agent can play a multi-armed bandit game(4 bandits). Figure 4.3 shows an impression of the game developed. Each bandit, when selected, generates a uniformly distributed random reward within the support mentioned as in Table 4.1.

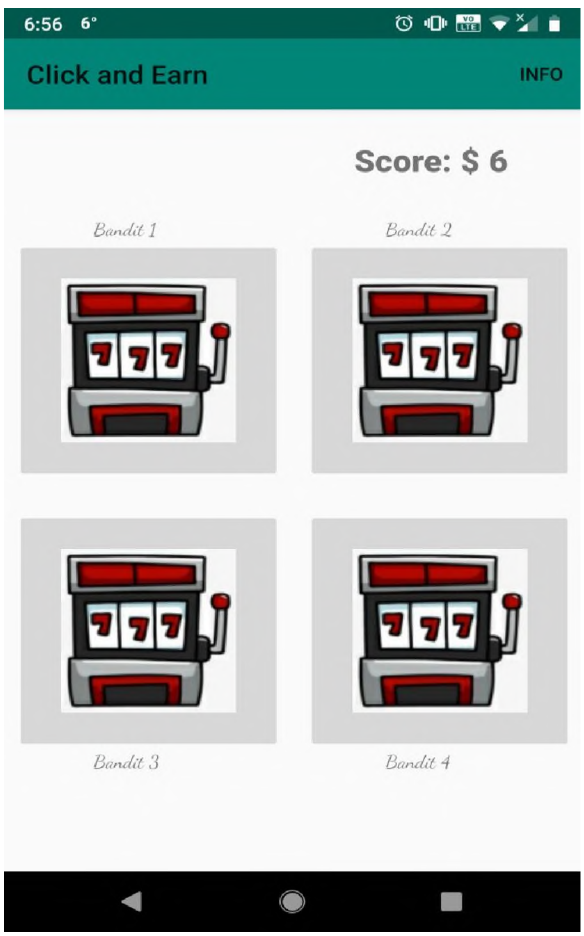

Figure 4.3. Multi-Armed Bandit ANDROID application

Table 4.1. Reward Distribution of Bandits

Bandit	Reward Distribution
1	U(1,20)
2	U(1,30)
3	U(1,40)
4	U(1,50)

We conducted a simple lab experiment where we asked players to play the ANDROID game which allows them to select a bandit to receive a random reward and can decide when they wanted to stop. They are informed that they will receive all the money won by them until the time they decide to stop. Also, we made sure that players are unaware of the bandits' reward distribution and do not discuss with each other while playing the game to avoid biases on the bandits.

Based on preliminary data collected using this application, our algorithm predicts the workers' stopping time with 26% accuracy if the utilities are defined as immediate rewards. However, in practice, people's utility functions are known to constitute preferences across multiple attributes. This motivated us to model the utility function as a multi-attribute function. The results are presented in the further sections.

4.2. MULTI-ATTRIBUTE UTILITY

In this Section, we will evaluate the algorithms proposed in Section 3.2 on data collected from simulation experiments. We present the experimental setup and the error performance details of our proposed algorithms.

4.2.1. Simulation Results for OL-BB, BL-QP, OL-QP Algorithms (Known Weights). To validate the algorithms proposed (in Section 3.2.2), when the utility of the crowd worker is a multi-attribute value with known weights, we designed a simulation experiment, where we assumed four tasks are available at the crowd worker $\mathcal{W}$ with equal probability. Each, when selected, produces m γ-distributed random rewards with shape

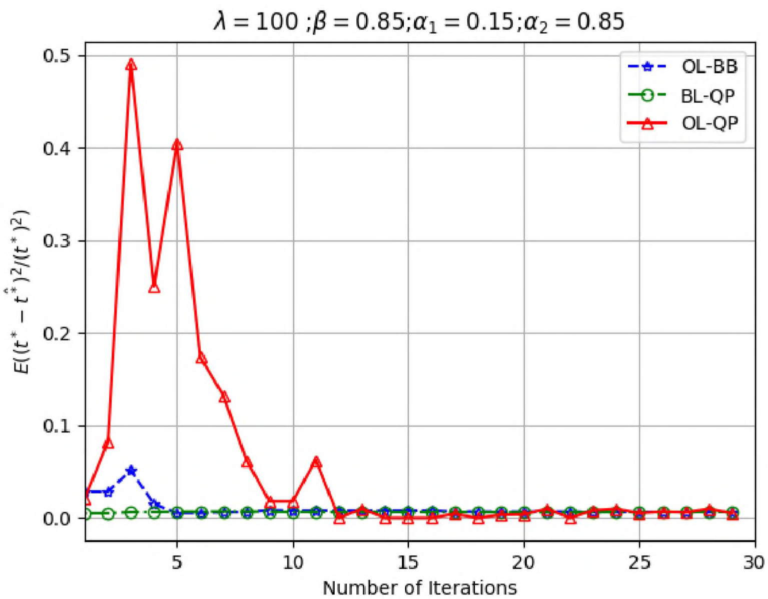

Figure 4.4. Error performance while predicting stopping time $\hat{t^*}$ when $\lambda = 100; \beta = 0.85; \alpha_1 = 0.15; \alpha_2 = 0.85$

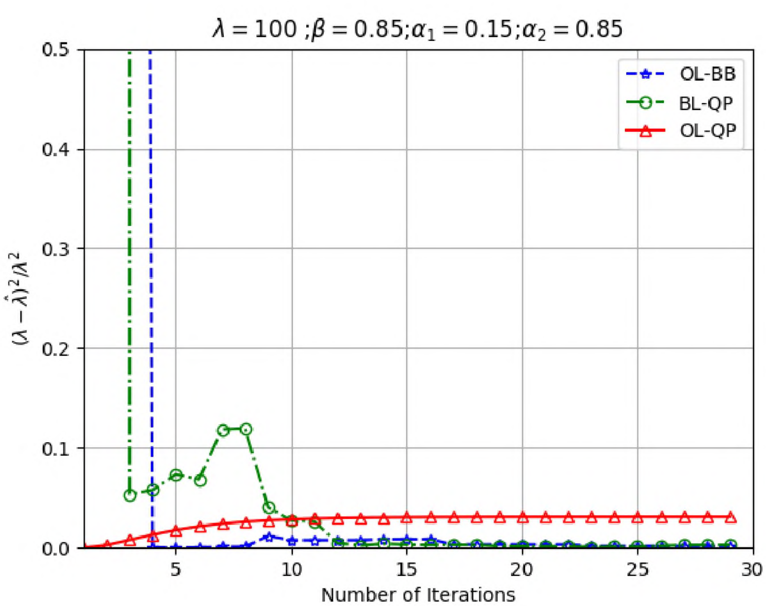

Figure 4.5. Error performance while estimating model parameter $\hat{\lambda}$ when $\lambda = 100, \beta = 0.85; \alpha_1 = 0.15; \alpha_2 = 0.85$

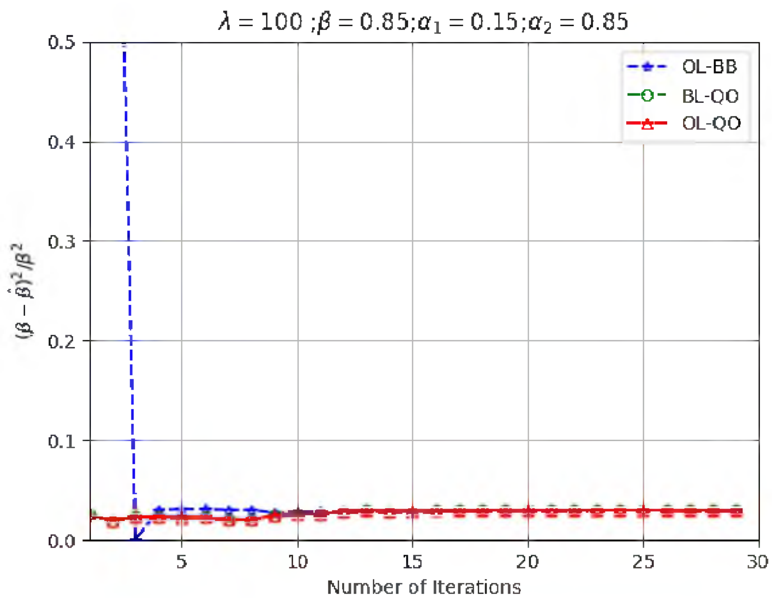

Figure 4.6. Error performance while estimating model parameter $\hat{\beta}$ when $\lambda = 100, \beta = 0.85; \alpha_1 = 0.15; \alpha_2 = 0.85$

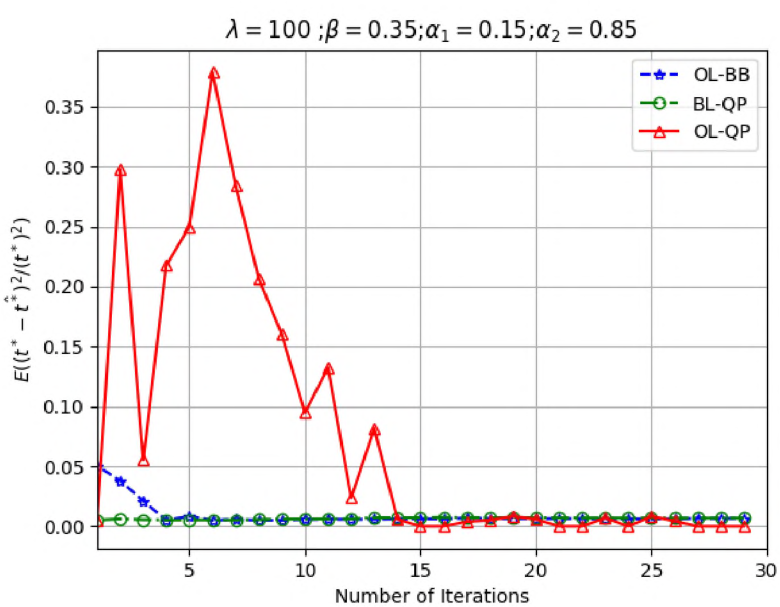

Figure 4.7. Error performance while predicting stopping time $\hat{t^*}$ when $\lambda = 100, \beta = 0.35; \alpha_1 = 0.15; \alpha_2 = 0.85$

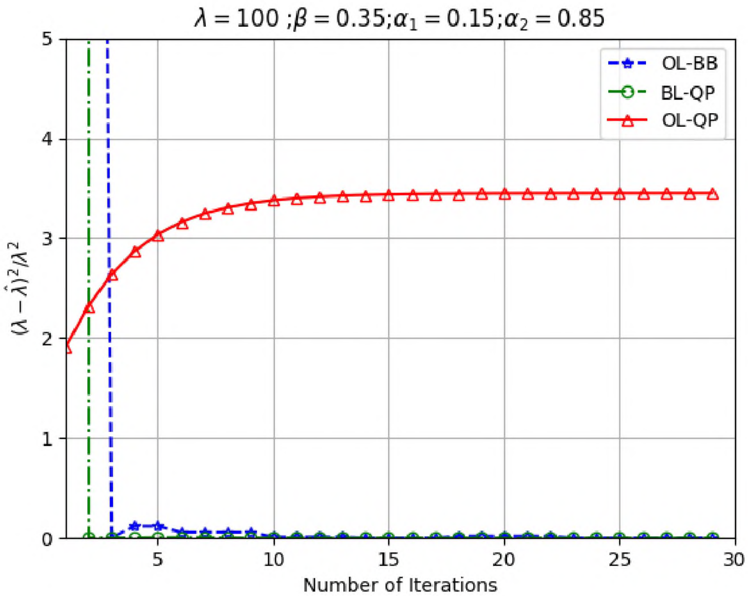

Figure 4.8. Error performance while estimating model parameter $\hat{\lambda}$ when $\lambda = 100, \beta = 0.35; \alpha_1 = 0.15; \alpha_2 = 0.85$

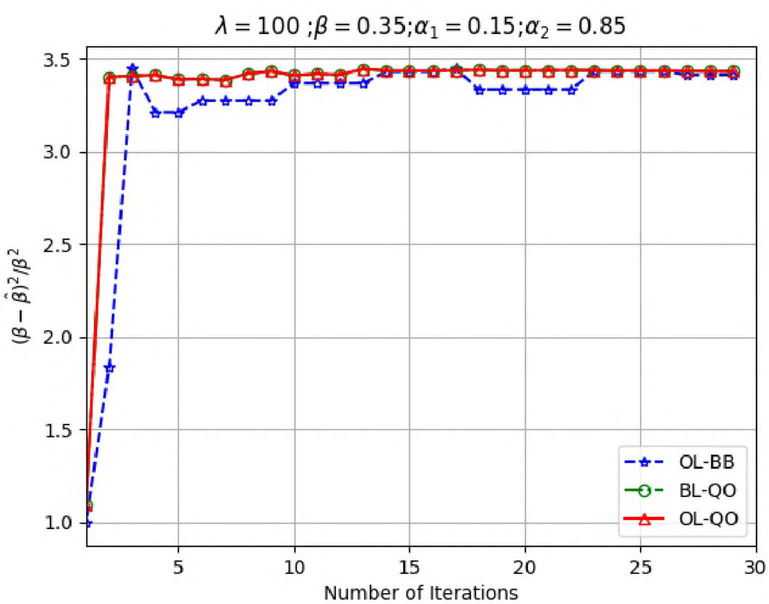

Figure 4.9. Error performance while estimating model parameter $\hat{\beta}$ when $\lambda = 100, \beta = 0.35; \alpha_1 = 0.15; \alpha_2 = 0.85$

parameter ($\alpha = 2$) and scale parameter ($\beta = 4$) corresponding to m attributes of the task. We also assume that the worker evaluates each task based on two ($m = 2$) attributes with known weights $\alpha_1 = 0.15, \alpha_2 = 0.85$ respectively. The regularization tuning parameter used in online learning algorithm (OL-QP) is assumed to be $\eta = 1000$. Assuming that the worker's model parameters are $\lambda = 100, \beta = 0.85$ in the first case and $\lambda = 100; \beta = 0.35$ in the second case, we ran several Monte-Carlo simulations of the proposed three algorithms and estimated model parameters $\hat{\lambda}, \hat{\beta}$. Using the estimated model parameters and stopping time Equation (3.9), we predicted the stopping time $\hat{t}^*$ of the crowdworker.

In the simulation results, we observed that average normalized error of the predicted stopping time using all the three proposed algorithms converges to zero consistently. This can be seen in the Figure 4.4 and Figure 4.7. When $\beta > 0.5$, we can see that all the three algorithms estimates the model parameters accurately and error converges to zero (shown in Figure 4.5 and Figure 4.6). Although, we can see convergence in the estimation error when $\beta < 0.5$ (shown in Figure 4.9 and 4.8), the error is quite high when compared to the former case. Nevertheless, this does not have any effect on the prediction of stopping time. From the Figure 4.7, it is evident that all the three algorithms perform well in predicting stopping time in the case of ($\beta < 0.5$) even though the β estimation error is high.

4.2.2. Simulation Results for BL-AM Algorithm (Unknown Weights) . In order to validate the algorithms proposed in Section 3.2.3 when the utility function is multi-attribute function with unknown weights, we designed a simulation experiment, where we assumed that 4 tasks are available at the crowd worker $\mathcal{W}$ with equal probability. Each task, when selected, produces m γ-distributed random rewards with shape parameter ($\alpha = 2$) and scale parameter ($\beta = 4$) corresponding to m attributes of the task. We also assume that the worker evaluates each task based on two ($m = 2$) attributes with unknown weights. Assuming that the worker's model parameters are ($\lambda = 100, \beta = 0.85$) in the first case and ($\lambda = 100; \beta = 0.35$) in the second case, we ran several Monte-Carlo simulations of

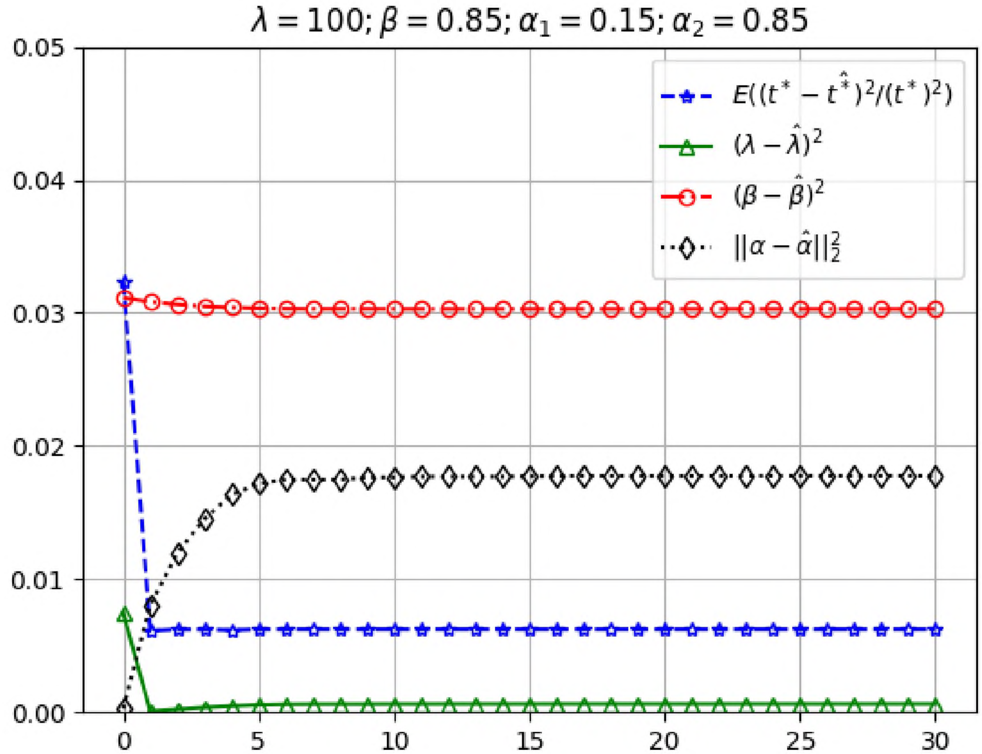

Figure 4.10. Error performance in predicting stopping time $\hat{t^*}$ and estimating $\hat{\lambda}, \hat{\beta}, \hat{\alpha}$ when $\lambda = 100, \beta = 0.35; \alpha_1 = 0.15; \alpha_2 = 0.85$

the proposed algorithm and estimated model parameters $(\hat{\lambda}, \hat{\beta})$. Using the estimated model parameters and stopping time Equation (3.16), we predicted the stopping time $(\hat{t^*})$ of the crowd worker.

From the Figures 4.10 and 4.11, it is evident that the proposed batch learning algorithm predicts the stopping time accurately. When $\beta > 0.5$, we can see that the estimation error with respect to model parameters (λ, β, α) is converging to zero as the number of iterations increase (shown in Figure 4.10). However, in the case of $\beta < 0.5$, we observe that the estimation error concerning β parameter is quite high compared to

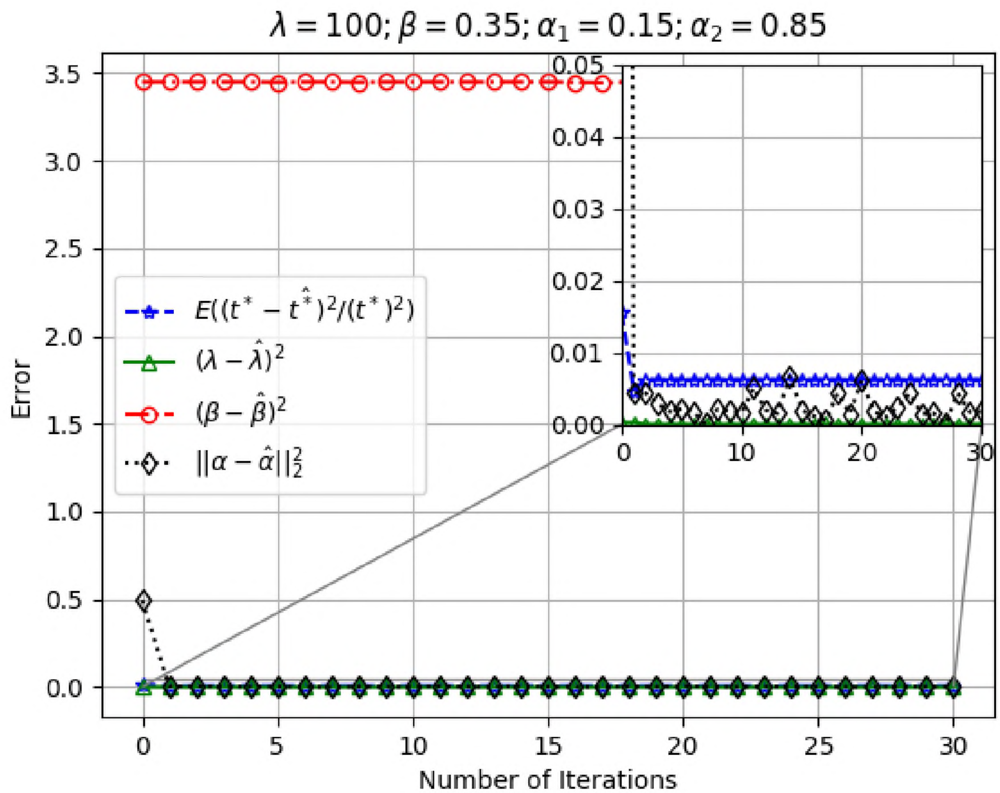

Figure 4.11. Convergence of error incase of multi-attribute utility with unknown weights($\lambda = 100; \beta = 0.35; \alpha_1 = 0.15; \alpha_2 = 0.85$)

the former case. However, we observed that this error does not have much impact on the prediction of stopping time. Figure 4.11 shows that the normalized error of the prediction stopping time is converging to zero as the number of iterations increases.

4.3. CROWDSOURCING PLATFORM DEVELOPMENT

In order to validate our proposed algorithms in the case of multi-attribute utility on real-data, we need a platform that can give us information about the rewards associated with the task's multiple attributes. Existing platforms like Amazon MTurk or Crowd Flower does not provide this information due to privacy concerns. Hence, we developed a web application in our lab that replicates a crowdsourcing platform and captures information about other attributes of the task apart from monetary rewards. This platform allows the crowd workers to create a profile for themselves, perform micro-tasks, and earn money for the executed tasks. It supports three types of tasks, such as (i) Handwriting Recognition, where the digit in the image has to be labeled (ii) Image Labeling, where the object in the image has to be identified and labeled and (iii) Content Moderation, where tweets need to labeled as sensitive/insensitive. Figures 4.12 and 4.13 shows sample task selection in the developed application.

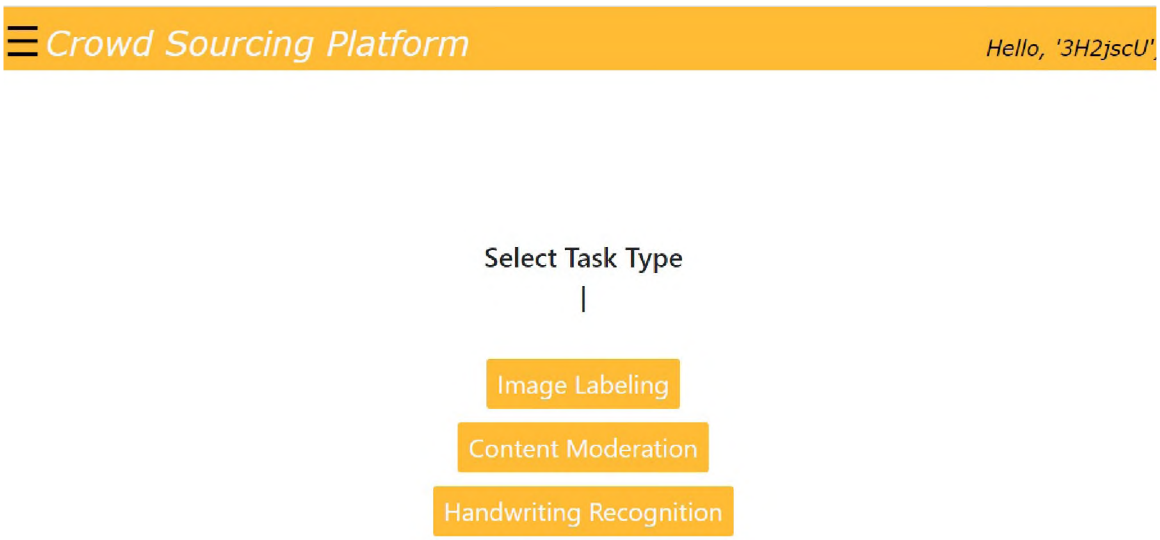

Figure 4.12. Crowdsourcing Platform: Sample type of task selection

Once the worker chooses the type of task he/she wants to execute, they are presented with a list of tasks available along with the details like expected time to complete the task, skills required, the monetary reward he/she receives after the successful completion of the task and the employer information. The worker can thoroughly examine the available tasks and then choose one to execute. We have created the tasks using publicly available datasets such as the MNIST handwriting recognition dataset(60000 Images), Fruits 360 dataset(67692 Images), and Hate-speech dataset from the white supremacy forum(10,568 sentences) [54]. Figures 4.15,4.14,4.16 shows the sample tasks for each type.

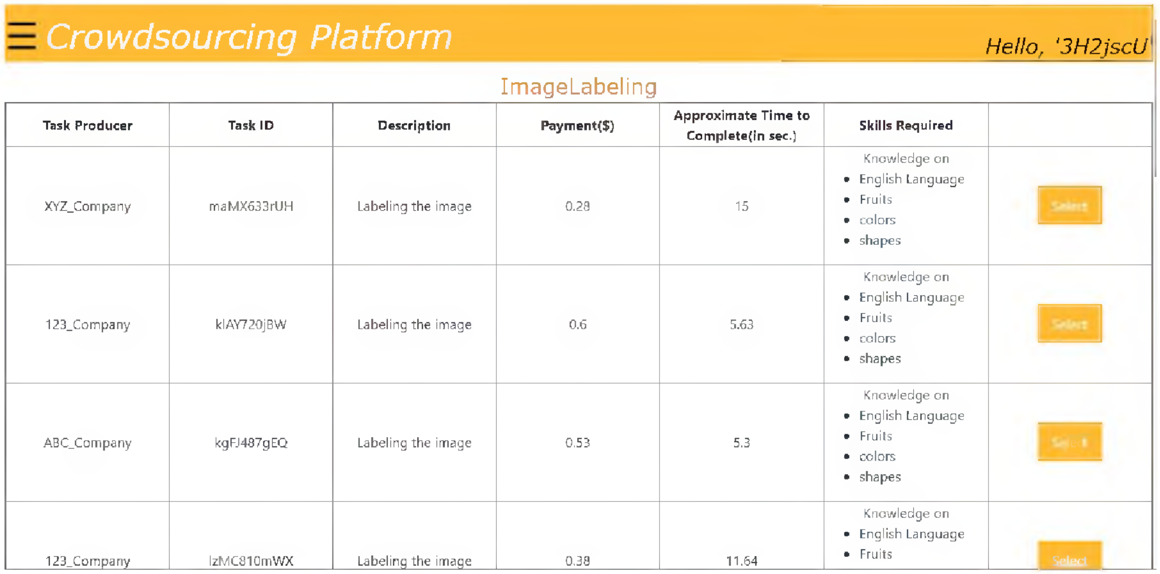

Figure 4.13. Crowdsourcing Platform: Sample task selection for execution

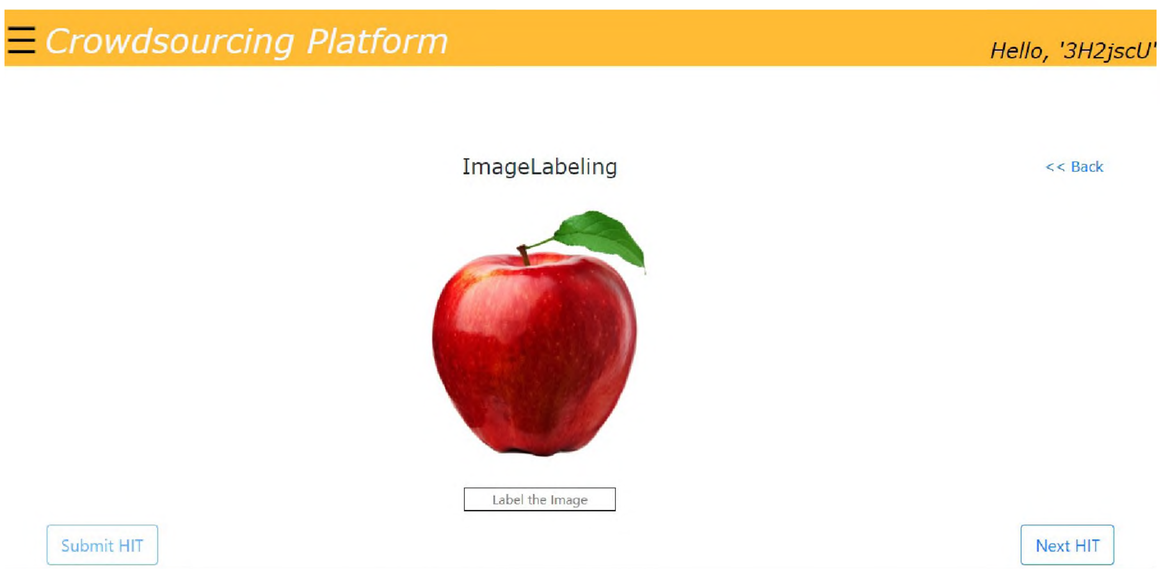

Figure 4.14. Crowdsourcing Platform: Sample Image Labeling task

The process of data collection using this developed web application is still under process, and the performance of the proposed algorithms on the real data is yet to be evaluated.

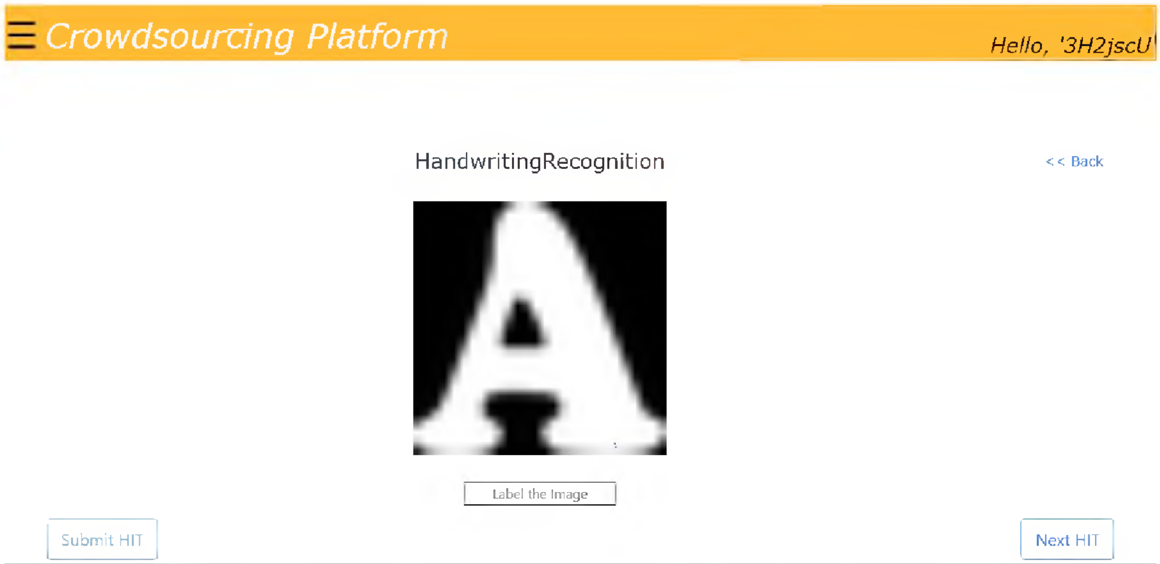

Figure 4.15. Crowdsourcing Platform: Sample Handwriting Recognition task

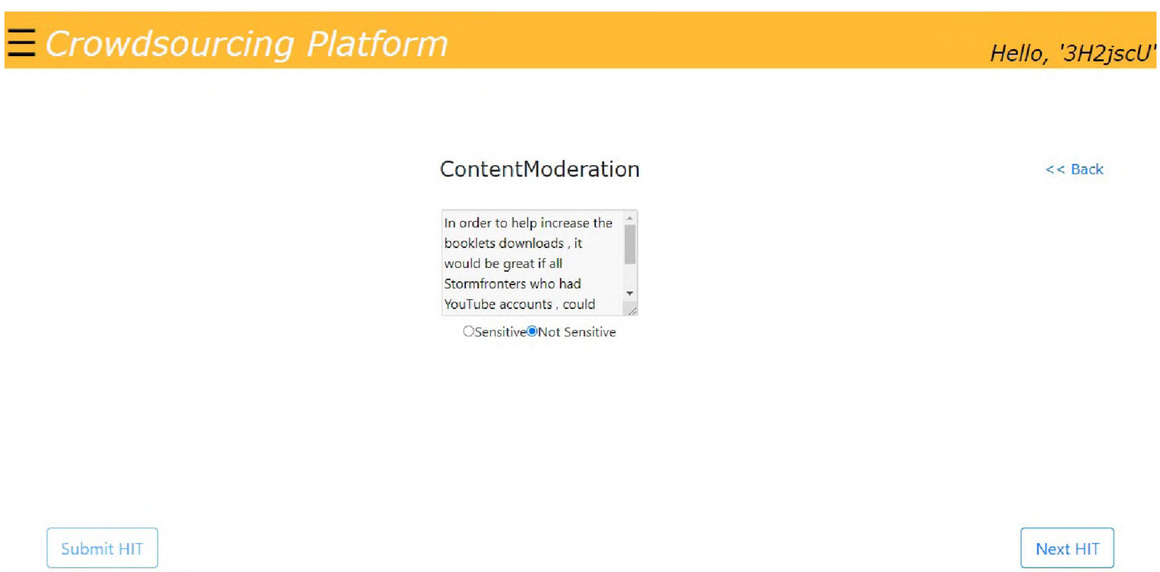

Figure 4.16. Crowdsourcing Platform: Sample Content Moderation task

5. CONCLUSION AND FUTURE WORK

In summary, we have proposed a novel heuristic called *Discounted Satisficing* to model stopping time of the human agent during sequential decision-making. Algorithms were presented to predict the stopping time of the agent exhibiting discounted satisficing heuristic and their performance on simulation data was also shown. The crowdsourcing platform designed to capture rewards with respect to multiple attributes of the executed task was also presented. Although our algorithms fails to estimate model parameters accurately in the case of human agents with greater discontent levels; it predicts stopping accurately. This led us to prove that two human agents with different preference order over attributes, thresholds and discontent levels can have same stopping time in the presence of fixed set of rewards.

Our next step is to collect real decisions of human agents using the developed web application and analyze our algorithms' performance on the real data. In addition, we would also like to extend our work by relaxing some of our assumptions such as immediate rewards of the tasks are perfectly observable, and agents always experience a diminishing threshold, especially since the above assumptions may not hold during every sequential decision-making situation in reality. We would also like to extend our model to settings where people adjust their threshold levels on-the-fly because of sudden surprise in the dynamic feedback from their instantaneous decisions.

REFERENCES

[1] M. Roohnavazfar, D. Manerba, J. C. De Martin, and R. Tadei, "Optimal paths in multi-stage stochastic decision networks," *Operations Research Perspectives*, vol. 6, p. 100124, 2019.

[2] Y. Wu and D. H. Tsang, "Dynamic rate allocation, routing and spectrum sharing for multi-hop cognitive radio networks," in *2009 IEEE International Conference on Communications Workshops*, pp. 1–6, IEEE, 2009.

[3] T. Schulze, S. Krug, and M. Schader, "Workers' Task Choice in Crowdsourcing and Human Computation Markets," 2012.

[4] L. B. Chilton, J. J. Horton, R. C. Miller, and S. Azenkot, "Task Search in a Human Computation Market," in *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '10, (New York, NY, USA), pp. 1–9, ACM, 2010.

[5] M. Yuen, I. King, and K. Leung, "Task Matching in Crowdsourcing," in *2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing*, pp. 409–412, Oct 2011.

[6] J. Von Neumann and O. Morgenstern, "Theory of games and economic behavior, 2nd rev," 1947.

[7] H. A. Simon, "Models of man; social and rational.," 1957.

[8] J. G. M. H. A. Simon, *Organizations*. 1958.

[9] S. Danziger, J. Levav, and L. Avnaim-Pesso, "Extraneous factors in judicial decisions," *Proceedings of the National Academy of Sciences*, vol. 108, no. 17, pp. 6889–6892, 2011.

[10] S. S. Iyengar and M. R. Lepper, "When choice is demotivating: Can one desire too much of a good thing?," *Journal of personality and social psychology*, vol. 79, no. 6, p. 995, 2000.

[11] C. Newton, "Facebook will pay $52 million in settlement with moderators who developed ptsd on the job." `https://www.theverge.com/2020/5/12/21255870/facebook-content-moderator-settlement-scola-ptsd-mental-health`, 2020.

[12] D. Gupta, I. Sen, N. Sachdeva, P. Kumaraguru, and A. B. Buduru, "Empowering first responders through automated multimodal content moderation," in *2018 IEEE International Conference on Cognitive Computing (ICCC)*, pp. 1–8, 2018.

[13] “Content moderation:the future is bionic.” https://www.accenture.com/cz-en/_acnmedia/PDF-47/Accenture-Webscale-New-Content-Moderation-POV.pdf, 2017.

[14] S. Karunakaran and R. Ramakrishnan, “Testing stylistic interventions to reduce emotional impact in content moderation workers,” 2019.

[15] F. P. de Lange, O. Jensen, and S. Dehaene, “Accumulation of evidence during sequential decision making: The importance of top–down factors,” *Journal of Neuroscience*, vol. 30, no. 2, pp. 731–738, 2010.

[16] L. K. Fellows, “The cognitive neuroscience of human decision making: a review and conceptual framework,” *Behavioral and cognitive neuroscience reviews*, vol. 3, no. 3, pp. 159–172, 2004.

[17] A. Wald and J. Wolfowitz, “Bayes solutions of sequential decision problems,” *The Annals of Mathematical Statistics*, vol. 21, no. 1, pp. 82–99, 1950.

[18] B. U. Forstmann, R. Ratcliff, and E.-J. Wagenmakers, “Sequential sampling models in cognitive neuroscience: Advantages, applications, and extensions,” *Annual review of psychology*, vol. 67, pp. 641–666, 2016.

[19] S. J. Gershman, A. B. Markman, and A. R. Otto, “Retrospective revaluation in sequential decision making: A tale of two systems.,” *Journal of Experimental Psychology: General*, vol. 143, no. 1, p. 182, 2014.

[20] W. R. Thompson, “On the likelihood that one unknown probability exceeds another in view of the evidence of two samples,” *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[21] R. N. Bradt, S. Johnson, and S. Karlin, “On sequential designs for maximizing the sum of n observations,” *The Annals of Mathematical Statistics*, vol. 27, no. 4, pp. 1060–1074, 1956.

[22] W. Vogel, “A sequential design for the two armed bandit,” *The Annals of Mathematical Statistics*, vol. 31, no. 2, pp. 430–443, 1960.

[23] D. Feldman *et al.*, “Contributions to the" two-armed bandit" problem,” *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 847–856, 1962.

[24] J. Gittins, “A dynamic allocation index for the sequential design of experiments,” *Progress in statistics*, pp. 241–266, 1974.

[25] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction.* MIT press, 2018.

[26] J. I. Gold and M. N. Shadlen, “The neural basis of decision making,” *Annual review of neuroscience*, vol. 30, 2007.

[27] P. Dayan and N. D. Daw, "Decision theory, reinforcement learning, and the brain," *Cognitive, Affective, & Behavioral Neuroscience*, vol. 8, no. 4, pp. 429–453, 2008.

[28] M. Furman and X.-J. Wang, "Similarity effect and optimal control of multiple-choice decision making," *Neuron*, vol. 60, no. 6, pp. 1153–1168, 2008.

[29] D. E. Acuña and P. Schrater, "Structure learning in human sequential decision-making," *PLoS computational biology*, vol. 6, no. 12, 2010.

[30] P. B. Reverdy, V. Srivastava, and N. E. Leonard, "Modeling human decision making in generalized gaussian multiarmed bandits," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 544–571, 2014.

[31] M. I. Rabinovich, R. Huerta, and V. Afraimovich, "Dynamics of sequential decision making," *Physical review letters*, vol. 97, no. 18, p. 188103, 2006.

[32] E. M. Tartaglia, A. M. Clarke, and M. H. Herzog, "What to choose next? a paradigm for testing human sequential decision making," *Frontiers in psychology*, vol. 8, p. 312, 2017.

[33] Y. S. Chow, H. Robbins, and D. Siegmund, *The theory of optimal stopping*. Dover, 1991.

[34] T. S. Ferguson, "Optimal stopping and applications," 2006.

[35] K. Kolomvatsos, C. Anagnostopoulos, and S. Hadjiefthymiades, "An efficient recommendation system based on the optimal stopping theory," *Expert Systems with Applications*, vol. 41, no. 15, pp. 6796 – 6806, 2014.

[36] X. S. Wang, I. Cruz, A. Delis, and G. Huang, eds., *On the Use of Optimal Stopping Theory for Improving Cache Consistency*, (Berlin, Heidelberg), Springer Berlin Heidelberg, 2012.

[37] N. Morali and R. Soyer, "Optimal stopping in software testing," *Naval Research Logistics (NRL)*, vol. 50, no. 1, pp. 88–104, 2003.

[38] R. Dong, E. Miehling, and C. Langbort, "Protecting consumers against personalized pricing: A stopping time approach," *arXiv preprint arXiv:2002.05346*, 2020.

[39] D. BERNOULLI, "Exposition of a new theory on the measurement," *Econometrica*, vol. 22, pp. 23–36, 1954.

[40] A. P. Wierzbicki, "A mathematical basis for satisficing decision making," *Mathematical Modelling*, vol. 3, no. 5, pp. 391 – 405, 1982. Special IIASA Issue.

[41] B. Schwartz, A. Ward, J. Monterosso, S. Lyubomirsky, K. White, and D. R. Lehman, "Maximizing versus satisficing: Happiness is a matter of choice.," *Journal of personality and social psychology*, vol. 83, no. 5, p. 1178, 2002.

[42] A. M. Parker, W. B. De Bruin, and B. Fischhoff, "Maximizers versus satisficers: Decision-making styles, competence, and outcomes," *Judgment and Decision making*, vol. 2, no. 6, p. 342, 2007.

[43] E. F. Harrison and M. A. Pelletier, "Managerial attitudes towards strategic decisions: Maximizing versus satisficing outcomes," *Management Decision*, vol. 35, no. 5, pp. 358–364, 1997.

[44] A. Roets, B. Schwartz, and Y. Guan, "The tyranny of choice: A cross-cultural investigation of maximizing-satisficing effects on well-being," *Judgment and Decision Making*, vol. 7, no. 6, p. 689, 2012.

[45] O. Kaminska, A. L. McCutcheon, and J. Billiet, "Satisficing among reluctant respondents in a cross-national context," *Public Opinion Quarterly*, vol. 74, no. 5, pp. 956–984, 2010.

[46] A. Kapelner and D. Chandler, "Preventing Satisficing in Online Surveys," *Proceedings of CrowdConf*, 2010.

[47] P. Reverdy, V. Srivastava, and N. E. Leonard, "Satisficing in multi-armed bandit problems," *IEEE Transactions on Automatic Control*, vol. 62, pp. 3788–3803, Aug 2017.

[48] D. I. Ignatov, A. Kaminskaya, N. Konstantinova, and A. Konstantinov, "Recommender System for Crowdsourcing Platform Witology," in *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 1, pp. 327–335, Aug 2014.

[49] S. Schnitzer, *Task Recommendation in Crowdsourcing Platforms*. PhD thesis, Technische Universität, 2019.

[50] K. Hata, R. Krishna, L. Fei-Fei, and M. S. Bernstein, "A Glimpse Far into the Future: Understanding Long-term Crowd Worker Quality," in *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, (New York, NY, USA), pp. 889–901, ACM, 2017.

[51] V. S. S. N. Mounica Devaguptapu, "Predicting crowd worker's stopping time based on discounted satisficing." `https://www.humancomputation.com/assets/papers/146.pdf`, 2019.

[52] I. E. Sutherland and G. W. Hodgman, "Reentrant Polygon Clipping," *Commun. ACM*, vol. 17, pp. 32–42, Jan. 1974.

[53] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[54] O. de Gibert, N. Perez, A. García-Pablos, and M. Cuadros, "Hate speech dataset from a white supremacy forum," in *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, (Brussels, Belgium), pp. 11–20, Association for Computational Linguistics, Oct. 2018.

VITA

Mounica Devaguptapu is an graduate research assistant under Dr. Venkata Sriram Siddhardh Nadendla in the Computer Science department at Missouri University of Science and Technology (in short, Missouri S&T). She received her Master's degree from Missouri S&T in August 2020 and her Bachelor's degree from Jawaharlal Nehru Technological University (India) in June 2014. Before joining Missouri S&T, she worked as a Software Developer in Tata Consultancy Services Ltd., India (in short, TCS) during 2014-18. During her TCS tenure, she received *'Star Performer of the Month'* award twice for her efforts in resolving customer trouble reports promptly.