ABSTRACT

A MODIFIED Q-LEARNING APPROACH FOR PREDICTING MORTALITY IN PATIENTS DIAGNOSED WITH SEPSIS

by Noah Michael Dunn

Among the medical crises of modern times, medically-diagnosed Sepsis persists as an ongoing condition yielding high mortality across all spectra of patients. Patients with Sepsis suffer variable symptoms, making it hard to evaluate severity, and the outcome of patients who develop Sepsis can range from full-recovery to death. The FOOTON model proposed in this study (named for its use of the qSOFA and SOFA scoring metrics) seeks to aid physicians in evaluating patient condition severity. The FOOTON model makes use of a stratified, cross-validated version of data provided by the MIMIC-III dataset to construct four variants of a binary classification model. Upon the completion of the model construction, the models can take patient data as input and they will output their prediction on the binary outcome of the patient (life or death). Of the four models produced, two models were shown to have the most promise. The imbalanced, unweighted variant of the model which performs at an average accuracy of 78.413% overall, and a balanced weighted variant, which performs at an average accuracy of 61.638% for dead patients. Providing this capability as an assistive tool for physicians can allow for the prioritization of limited resources to individuals at a greater risk of dying, with the potential to decrease overall patient mortality.

A MODIFIED Q-LEARNING APPROACH FOR PREDICTING MORTALITY IN PATIENTS DIAGNOSED WITH SEPSIS

Thesis

Submitted to the

Faculty of Miami University

in partial fulfillment of

the requirements for the degree of

Masters of Computer Science

by

Noah Michael Dunn

Miami University

Oxford, Ohio

2021

Advisor: Dhananjai M. Rao, Ph. D

Reader: Philippe J. Giabbanelli, Ph. D

Reader: Vaskar Raychoudhury, Ph. D

This Thesis Titled

A MODIFIED Q-LEARNING APPROACH FOR PREDICTING MORTALITY IN PATIENTS DIAGNOSED WITH SEPSIS

by

Noah Michael Dunn

has been approved for publication by

The College of Engineering and Computing

and

The Department of Computer Science and Software Engineering

Dhananjai M. Rao, Ph. D

Philippe J. Giabbanelli, Ph. D

Vaskar Raychoudhury, Ph. D

Contents

List of Tables

List of Figures

Acknowledgements

First and foremost above all else, I hold Jesus Christ, the author and finisher of my faith, God almighty, as the chief author of this document. Without him, I am nothing, and this document ceases to exist. All honor, praise, and glory to him.

I would like to thank Dr. Dhananjai Rao for advising this research project. He has gone beyond his call of duty, acting as a mentor, a colleague, and a friend. I would additionally like to thank Dr. Philippe Giabbanelli for his expertise in the field of machine learning. He never strayed from a question and always provided resources to help me grow. I would like to thank Dr. Vaskar Raychoudhury for his role as a reader and mentor, always offering helpful advice on research flow, interpreting papers, and sources to draw inspiration from.

I would be remiss if I did not include a thank you to my parents, my grandparents, and my entire family for being able to help me become the student I am today. Their years of supporting my ambitions to learn and grow are the reason that this paper exists before you. In addition, my friends are a part of my extended family and even in the darkest of times, they have provided brevity to which no candle can hold a light. Ken Dillard and Ricky Cotton have been spiritual beacons throughout my entire college career, and it is to them I owe my continued spiritual growth and encouragement in my Christian walk.

The College of Engineering at Miami University and specifically the faculty of the Computer Science and Software Engineering Department have given me the resources to be able to succeed on this effort, and to them I owe a great thanks. Patty Strecker, the administrative assistant to the Computer Science and Software Engineering department, was vital in allowing me to schedule the events needed to fulfill my Master's degree. More importantly, she has provided years of encouragement and emotional support for my career at Miami University, of which I can never repay. Dr. Daniela Inclezan and Dr. Eric Bachmann served as heads of the graduate department during my time in the Master's program, and I owe them a great deal for their assurance of my schedule and graduation requirement.

Last but not least, I would like to thank Dr. Md. Osman Gani. Serving as my original advisor, and starting my journey on this research during my undergraduate career. He is someone I am proud to call one of many lifelong mentors and friends. This piece of work would not have existed without him.

Dear reader, I thank you. May God Bless you,

Noah Michael Dunn

Chapter 1

Introduction

1.1 Motivation

In the age of the big data revolution, machine learning and advanced computational statistics dominate as a means of understanding the significance of large batches of data [1]. Domains outside of computer science are relying on an understanding of complex algorithms to improve the quality of life across sectors, and this in particular is impacting modern medical research. Hospitals and particularly Intensive Care Units (ICUs) are often crowded with more patients than they are capable of handling in a reasonable amount of time [2]. In cases of serious medical emergencies, time not spent working directly with a patient can result in more dangerous complications, and even death. As a result, many researchers are exploring the domain of computer-assisted diagnosis to gain as much information on patients as quickly as possible [3]. Doctors that are provided even a glimpse at the full scope of their patients' circumstances and conditions in the form of computer-generated predictions will be able to make more accurate judgement calls on distribution of available resources to patients most in need.

1.2 Contributions

The usage of computer-aided modeling, or more specifically machine-learning related modeling to diagnose patients, or predict patient outcomes, is not a new idea. Specifically, the research here is intended to contribute to the existing knowledge base in the following ways:

1. Build a model that makes use of Q-Learning techniques and patient data to predict patient

mortality.

2. Optimize that model using known applied statistical best practices and intuitions from the existent literature on Sepsis

Chapter 2

Background & Related Work

2.1 Sepsis

2.1.1 Defining and Understanding Sepsis

According to the Third International Consensus Definitions for Sepsis and Sepsis Shock, commonly referred to as “Sepsis-3", *Sepsis* is defined as “life threatening organ dysfunction caused by a dysregulated host response to infection [4]."In layman’s terms, Sepsis is the “body’s extreme response to an infection" which often occurs when “an infection you already have – in your lungs, urinary tract, or somewhere else – triggers a chain reaction throughout your body [5]."

Sepsis has received attention for the past 30 years more so than ever as it is quickly being identified as one of the “largest causes of health loss worldwide" [6]. In 2017 alone, Sepsis totalled *48.9 million cases* worldwide and accounted for more than *11.0 million deaths across the globe* (with a 95% confidence interval) [6]. Because of this, Sepsis stands ranked in the top 10 causes of death worldwide [7]. Needless to say, now more than ever, Sepsis has been identified as a health concern worth investigating.

2.1.2 Septic Shock

An extreme form of Sepsis is known as *Septic Shock* which occurs when the body-wide infection due to sepsis “...leads to dangerously low blood pressure..." [8]. In recent years, work on understanding the causes behind Septic Shock and various potential treatments have reached an all-time high. On a microbiological level, small *Pathogens*, which consist of any organism that “[cause] disease to [their]

host", initialize a series of communications among a host's cells [9] [10]. The communication chain prompts the host's cells to become inflammatory or get bigger, across the entirety of the host's body. As a result, in a very short amount of time, a large portion of a person's cells suffer inflammation, and all of these cells in combination deal severe damage to the person's bodily tissue [10].

The reason that Septic Shock poses such an issue is that "The time window for interventions is short..." and any supplied treatment must be delivered soon after the infection begins to spread [10]. Septic Shock is identified as being one of the most deadly forms of Sepsis, making up an average mortality rate of *50%* [11]. At an average of *2-3%* of patients who come into the ICU identified as having Septic Shock, this amounts to a significant number of deaths caused by the condition.

2.1.3 Symptoms of Sepsis and Risk Groups

For doctors, the apparent difficulty in the detection of Sepsis the ambiguity of the signs and symptoms associated with it. All the symptoms are based on the underlying pathogen that is infecting specific parts of the affected person's body. Additionally, early genetic testing in patients has been shown to yield contradictory results [12]. To make matters additionally complicated, brain function is often "deranged" due to Sepsis impairing brain functionality in affected patients, even ones with otherwise normal brain function.

In terms of risk assessment, there are several groups of people who are more at risk of getting Sepsis. The following is a short list provided by WebMD [13]:

- People who are currently suffering from or have suffered from HIV/AIDS or Cancer
- People who take immune-system suppressors like steroids or drugs used to prevent "rejection of transplanted organs"
- Very young babies (3 months old or younger)
- The elderly (particularly those suffering any other health issues)
- Anyone who has recently suffered a major injury
- People with diabetes

Affliction	Known Symptoms + Signs
Sepsis	A fever of above 101 °F A heart rate higher than 90 BPM Breathing rate higher than 20 breaths per minute Probable or Confirmed Infection
Severe Sepsis	Any of the previous and: Patches of discolored skin Decreased Urination Changes in mental ability Low platelet (blood clotting cells) count Problems breathing Abnormal heart functions Chills due to fall in body temperature Unconsciousness Extreme weakness
Septic Shock	Any of the previous and extremely low blood pressure

Table 2.1: Symptoms of the Various Stages of Sepsis [14]

If any of these kinds of patients display any of the symptoms associated with Sepsis, they will be given tests that can provide helpful information to a doctor. Among initial tests, which determine "changes in body temperature, leukocyte count, heart rate, blood pressure, and respiration rate" are used to detect inflammation in a patient's body [13]. Following this, a doctor may use a version of the SOFA score to determine the likelihood that a patient has Sepsis given their vitals.

2.1.4 SOFA Score

The original Sequential Organ Failure Assessment (SOFA) score was developed by the European Society of Intensive Care Medicine back in 1994 [15]. In general, this score is used to determine how in danger a patient is of organ failure, due to Sepsis-related complications. The score *ranges from zero to four for each of six different organ systems*. A zero on the SOFA score represents a normal functioning organ, and a four represents a very abnormal organ [16]. Although Sepsis is not directly determined from any given case of organ failure, "...mortality rate is directly related to the degree of organ dysfunction [16]." The authors of the SOFA score make a careful note to point out here, "...SOFA score is designed not to predict outcome but to describe a sequence of complications

in the critically ill.." and as such, the '...SOFA score does not compete with the existing severity indexes but complements them [16]." The score for a given patient can be calculated by the use of the Table 2.2, summing the values that apply to the patient for each row.

SOFA Score	**1**	**2**	**3**	**4**
Respiration PaO2/FIO2 (mm Hg)	<400	<300	<220	<100
SaO2/FIO2	221 - 301	142 - 220	67 - 141	<67
Coagulation Platelets x 10^3/mm^3	<150	<100	<50	<20
Liver Bilirubin (mg/dL)	1.2 - 1.9	2.0 - 5.9	6.0 - 11.9	>12.0
Cardiovascular Hypotension	MAP <70	Dopamine ≤ 5 or Any Dobutamine	Dopamine >5 or Norepinephrine ≤ 0.1	Dopamine >15 or Norepinephrine >0.1
Glascow Coma Score	13 - 14	10 - 12	6 - 9	<6
Renal Creatine (mg/dL) or Urine Output (mL/d)	1.2 - 1.9	2.0 - 3.4	3.5 - 4.9 or <500	≥ 5.0 or 200

Table 2.2: The Sequential Organ Failure Assessment Score (SOFA) [17]

2.1.5 qSOFA Score

The former SOFA score, albeit thorough in scope, does not lend itself to be performed in a quick manner. As a result, the quickSOFA score commonly known as the *qSOFA score* was developed at the Sepsis-3 conference discussed earlier [4]. The design of this score was to get a much quicker idea of whether or not a patient had Sepsis. As opposed to the default SOFA score, which has 6 different metrics of analysis, one for each organ, the qSOFA score offers just one range of scoring. The qSOFA score ranges from values of *zero to three points, where two points or more represents "...a greater risk of death or prolonged intensive care unit stay"* [18]. In qSOFA testing, a patient adds a point to their default score of zero if any of the following three criteria apply to them [18].

1. Patient has low blood pressure, defined as a Systolic Blood Pressure of ≤ 100 mm Hg [19]
2. Patient has a high respiratory rate, defined as ≥ 22 breaths/minute
3. Patient possesses an altered mentation (change in mental ability), defined as scoring < 15 on the Glasgow coma scale [20]

The qSOFA Score test requires three easy to obtain measures of a patient's health, and particularly in cases outside the ICU, "...the simple qSOFA model performed similarly to more complex models like SOFA...outside the ICU [18]." Figure 2.1 shows risk relative to qSOFA score.

Component Tested	Score
Eye Response	
Eyes open spontaneously	4
Eye opening to verbal command	3
Eye opening to pain	2
No eye opening	1
Motor Response	
Obeys Command	6
Localises Pain	5
Withdraws from Pain	4
Flexion Response to Pain	3
Extension Response to Pain	2
No Motor Response	1
Verbal Response	
Oriented	5
Confused	4
Inappropriate Words	3
Incomprehensible Sound	2
No Verbal Response	1

Table 2.3: The Glasgow Coma Scale [21]

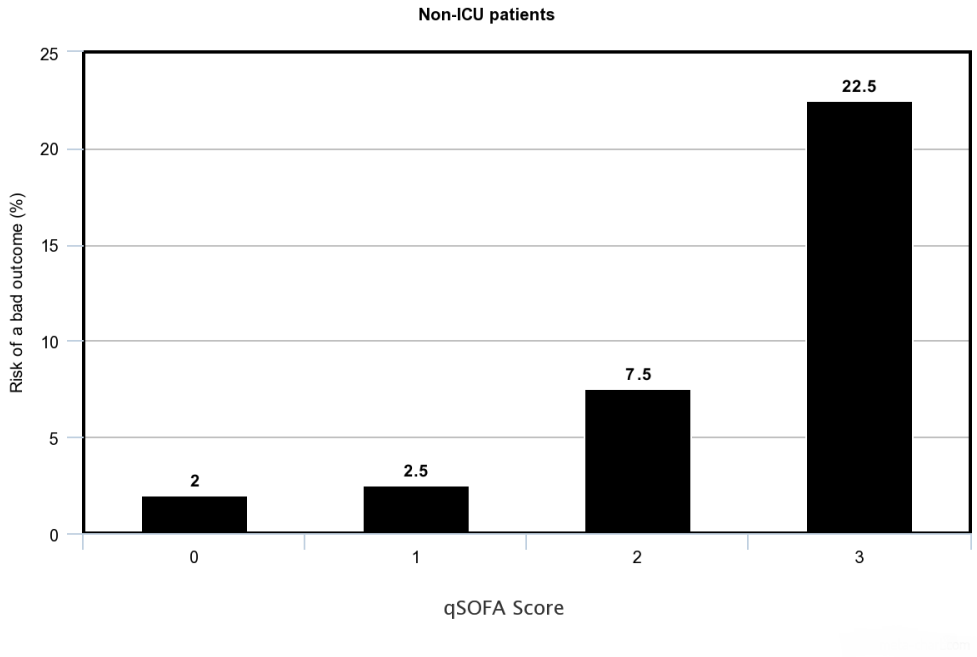


Figure 2.1: qSOFA Risk Assessment in a Non-ICU environment [18]

2.1.6 The Importance of Quick Treatment

After a patient is diagnosed, either by using a SOFA score, a qSOFA score, or some other metric, there is a very small window of time a patient has to receive treatment before their condition worsens quickly. For sepsis treatment, any "early goal directed therapy completed within the *first six hours of sepsis recognition* significantly decreases in-hospital mortality [22]." In a study conducted by Kumar et al., each hour of delay in treatment yielded an *average decrease in survival rate of 7.6%* [23]. This study also documents that the median time to effective treatment was six hours. In addition, the authors make note that time to effective treatment was the greatest predictor of patient outcome. Despite all of this, "only *50%* of septic shock patients received *effective antimicrobial therapy within 6 hours of documented hypotension*" (low blood pressure) [23]. A flowchart mapping the full understanding of Sepsis diagnosis and treatment is shown in *A.1*.

2.1.7 Understanding Sepsis Treatment

Quick treatment is a necessity for all potential Sepsis patients, and unlike the symptoms which are varied, the treatments are more streamlined. The Center for Disease Control CDC recommends patients who have been diagnosed with Sepsis to be treated in the following manner [24]:

- The patient should receive *antibiotics*
- Consistent blood flow should be provided to the patient's organs. This is often done through the provision of *Oxygen and Intravenous (IV) fluids to the organs*.
- The source of the infection should be treated

In a study conducted by Seymour et al., the researchers add that *quicker administration of antibiotics* were "..associated with low risk-adjusted in-hospital mortality", but quicker completion of IV fluids did not follow this trend. Duffy also offers a graph shown in Figure 2.2 to demonstrate the importance of early antibiotic administration in pediatric sepsis [25]. This graph also demonstrates how few patients get the treatment they need at the rate that they should.

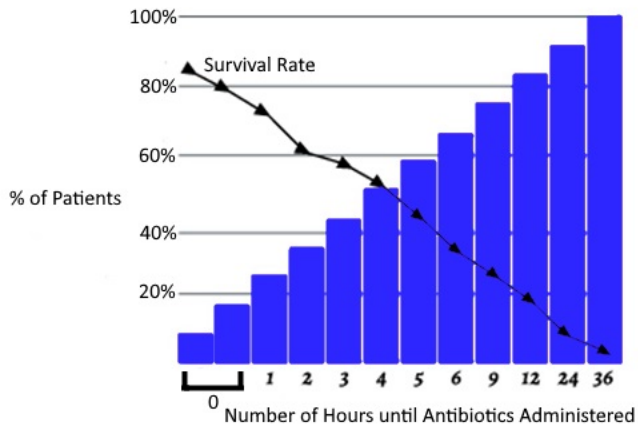


Figure 2.2: The Importance of Speed in Antibiotic Administration [25]

2.2 Q-Learning

The SOFA and qSOFA scores offer certain criteria that enable doctors to diagnose, with reasonable certainty, if a patient is suffering from Sepsis or not. However, it is possible that there are even better metrics to predict mortality due to Sepsis in a patient if provided enough specific information about a patient's vitals. Such ideas are what prompted researchers like Komorowski et al., to pursue the creation of an Artificial Intelligence (AI) clinician, as well as prompting PhysioNet's "Early Prediction of Sepsis from Clinical Data" challenge [26] [27]. Both of these are discussed in detail later on in *Section 2.4*. In the pursuit of manipulating known data to approximate outcomes, Machine Learning offers a worthy tool.

2.2.1 The Philosophy of Machine Learning

In his book, Machine Learning 2nd Edition, Marsland states that *"Machine learning ... is about making computers modify or adapt their actions ... so that these actions get more accurate, where accuracy is measured by how well the chosen actions reflect the correct ones* [28]." In the case of predicting Sepsis, the goal is to design a program that can take many patients' vitals information as data input, and output whether or not they have Sepsis with accuracy. Within the domain of Machine Learning, there are several different types of learning algorithms that can be used [28]:

- *Supervised Learning:* The algorithm is provided with a preset set of input data with known

labels (responses). The algorithm uses these to create a model that will give the correct answer for all the provided preset information. Also called “learning from exemplars".

- *Unsupervised Learning:* The algorithm is not provided with *labels (responses)*, but the algorithm attempts to create a model based on the similarities that the inputs have in common.
- *Reinforcement Learning:* The algorithm is told when it comes up with the incorrect response, but not why the response is incorrect. From this point, it tries multiple approaches until it gets the response correct. Also called “learning with a critic".

2.2.2 Reinforcement Learning

In particular for medical purposes, several research groups have used reinforcement learning for data that may produce results in a “sequence of states", such as data that deals with treatments or any data involving time [29] [30] [31]. In a recent study by Zhang et al. a form of reinforcement learning was used to optimize a treatment regiment for patients of drastically different characteristics [32]. Their approach, which fit linear models offers an easy interpretation (binary choices) and can be performed in most software groupings; however, it suffers when a linear model is chosen to map non-linear behavior. In studies of more generalized diagnosis, Ling et al. experimented with using entire descriptions of a patient’s information with Reinforcement Learning algorithms to diagnose. [33] Their approach uses Markov Decision Processes (MDPs) and a systematic processing of written English statements to attempt diagnosis on a patient. The research group also makes note of their use of the Deep Q-Learning Network approach, substituting a traditional value function with a approximation. Specific to the issue of diagnosing Sepsis, Komorowski et al. approached the problem using a combination of *Q-Learning and TD-learning*, expanded on further in the related works [26].

2.2.3 An Introduction to Q-Learning

In 1989, CJCH Watkins published a paper called “Learning from delayed rewards", which tackled the approach of previous reinforcement algorithms, but with a twist [34]. In his paper, Watkins

proposed only rewarding the algorithm if it accomplishes the goal it was oriented towards. Later, Watkins would go on to formalize his Machine Learning proposal under the name *"Q-Learning"* alongside several other researchers. While a more generalized algorithm would look for ways to improve the result from the initial test and response of a model, Q-Learning seeks to optimize individual *"qualities"* (this is where the Q comes from) of a model through incremental experimentation [34] [35]. Additionally, it is important to understand that Q-Learning is an *Off-Policy* algorithm. What this means is that at any given incremental step, the Q-Learning algorithm *does not* take into consideration *what previous path (policy)* was taken to achieve the desired outcome.

2.2.4 Markov Decision Processes

Since we are concerned exclusively with decisions made that do not take prior history into consideration, the underlying structure to any Q-Learning algorithm makes use of the mathematical construct of the MDP model [36]. Any given MDP contains the following [36]:

1. "A set of possible world states"
2. "A set of possible actions"
3. "A real valued reward function"
4. "A description of each action's effects in each state"

Also, all MDPs match Q-Learning in the idea that all decisions must be independent of decisions made in the past [37].

For practical benefit, David Silver offers an example MDP. In Silver's MDP, the set of all possible world states is all of the places a student could be: on Facebook, at the Pub, in any of the 3 classes, or at the "Pass state". The set of all possible actions is the collection of all *possible paths*. One path could be going to Classes 1 through 3 and then passing (which is our goal in this model). Alternatively, a student could go to the Pub, skip classes 1 and 2 and then go to class 3. The "real valued reward" in this case is how the decimal values present on the lines *accumulate*, giving higher reward weights to attending classes as opposed to visiting the pub. Step 4, the individual decimal

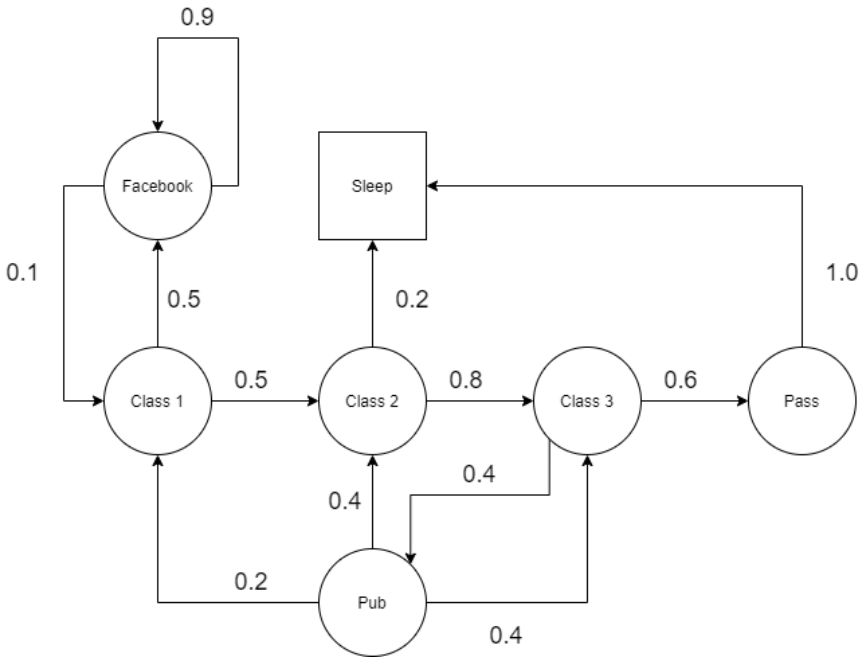


Figure 2.3: An MDP for Student Success Rate [37]

values act as our "description of each action's effects in each state", however in other models, this could be accompanied by an *explanation of the action at hand* [36] [37].

2.2.5 The Q-Equation and The Q-Learning Function

Understanding MDP is the foundation to understanding Q-Learning, as one of our objectives is to determine the best action in an MDP given a state in that MDP. However, the valued reward function described in the previous section is yet to be built. In Q-Learning, the process is to maximize the reward value at each iteration to achieve the optimal *"Q" or "Q-Function"*, using the *maximum Q-Value* at each step [38]. To start, we address the *Bellman equation* [39] [40]:

$$\boldsymbol{Q(s,a) = r + \gamma \max_{a'} Q\left(s', a'\right)} \tag{2.1}$$

The basic version of this equation states that the Q value ($Q(s,a)$) for the current state (s) and current action (a) is determined by the current reward (r) added to the maximum future reward/Q-Value ($Q(\boldsymbol{s', a'})$) for the next state ($\boldsymbol{s'}$) and action ($\boldsymbol{a'}$) multiplied by a discount factor ($\boldsymbol{\gamma}$). The discount

factor exists to ensure our algorithm prioritizes "...the discounted future reward at every step [39]."

This basic equation expands to the full *Q-Learning Equation* for determining reward values/Q-Values [39]:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha\left(r_{t+1} + \gamma \max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t)\right) \tag{2.2}$$

This is the same equation as before, with two modifications. The first is that this equation addresses the next Q-value ($Q_{t+1}(s_t, a_t)$) with respect to our current value Q-value ($Q_t(s_t, a_t)$). The second modification is the inclusion of the alpha (α), which represents "...the learning rate that controls how much the difference between previous and [the new] Q value is considered". [39] Given enough increments, our model will produce the *maximal Q-equation* which determines the *best action at every state* [39]. To reiterate, provided a state in an MDP, the *Q-Equation* will produce the best action from that state to get closer to the goal.

Symbol	Meaning
$Q_{t+1}(s_t, a_t)$	The next Q-Value
$Q_t(s_t, a_t)$	The current Q-Value
s_t	The current state
a_t	The current action
t	The current step
$t+1$	The next step
α	The learning rate, which determines the weight of the difference between the current value and the next
r_{t+1}	The next reward value
$\max_a Q_t(s_{t+1}, a) - Q_t(s_t, a_t)$	The maximal next Q-Value
γ	A discount factor to ensure that a future value is always selected

Table 2.4: The Symbols of the Q-Learning equation

2.2.6 Q-Learning in the Context of Mortality Prediction

For the sake of understanding the specific medical applications in Q-Learning, Yu C. et al. have surveyed the many different types of research taking place currently, all of which are using some

form of reinforcement learning, including variations of Q-Learning [41]. In another study specific to Sepsis, Gottesman et al. discuss guidelines to be used by reinforcement learning algorithms in the diagnosis of Sepsis, with a careful note to maintain "..caution and due diligence.." in implementation, as these are working with human lives [42]. Their discussion revolves mostly around the considerations that need to be made when researching with reinforcement algorithms. They also note the importance of providing as many details about a patient as possible to build a model, while also warning of confounding factors, or factors that affect one another. Q-Learning has been used already to diagnose Sepsis in patients, as seen in the Komorowski et al. study in detail later [26]. With the possibility of solving an MDP with two binary states/goals: *the patients lives or the patient dies*; Q-Learning has the potential to predict a given patient's outcome.

2.3 Statistical Modeling

Normally, Q-Learning relies on the creation of an MDP before solving. Sometimes, the individual designing an algorithm can spot and create an MDP naturally based on the context of the model, like in the case of the previous example *Figure 2.3*. In the case of diagnosing an illness, often the actions that lead from one diagnosis state to another are unclear. To avoid the random ordering and testing of MDP construction, we can employ some *Statistical Inference* to add a methodology behind optimization.

2.3.1 Statistical Inference

Within the past sixty years, statistical inference has dominated in many fields, including Computer Science, as the primary means of determining the significance of resulting data from an experiment [43]. In particular, statistical inference is used to "...[compare] particular statistics from one observational data set to another ... with an appropriate reference distribution to *judge the significance of those statistics* [44]." At its basis, the goal of statistical inference is to determine if there exists a relationship between some data, whether that relationship means anything, and how the knowledge of that relationship might translate to the theoretical "whole set of data" known as the population

[45]. For the problem of predicting mortality in Sepsis patients, we want to determine relationships among the available patient data, and create a model that represents certain characteristics of a given patient's data (like their blood pressure, body temperature, etc) as *predictors*, to determine if that patient will die or not [46]. This is done to build a model that represents the whole population, in this case, Sepsis patients.

2.3.2 Understanding SCM Scope

Structural Causal Models (SCMs) are a method of demonstrating and building models within the lens of statisical inference, but much like every other statistical method, some assumptions need to be addressed. A first concern is that the model builder is responsible for designating *exogenous variables* which are deemed outside the scope of the model, and *endogenous variables* which are inside the scope of the model. If a study is looking at the impact of burning various chemicals on the global environment, the researchers would deem several variables worth modeling as endogenous: temperature, population density, wildlife population. On the other hand, there are likely several variables that are not worth observing: GDP for an area, yearly fruit production. This *does not mean* that these factors do not have a relationship with a factor in the model, it simply means that the researcher has excluded them for the sake of scope [47].

2.3.3 Intervention

Particularly useful for medical applications, there are cases where through experimental trial runs, researchers may wish to inject "*Interventions*", where the researchers or something else modifies the conditions during the experiment to achieve the desired outcome [48] [49] [50]. In some cases, "...where randomized controlled experiments are not practical...", particularly in the cases of recorded data, intervention is still possible [47]. In an SCM, we can intervene by taking any given factor and fixing its value. Intervention offers the model-builder or experimenter the chance to substitute known values for values that a model would determine based on its findings. These are particularly useful when it is well known what impact a particular factor may have on an output. Fixing different factors' values and testing would produce models of varying quality, allowing a

researcher to potentially achieve a better overall model with intervention.

2.3.4 Applications for Mortality Prediction

Q-Learning and MDPs work together in the family of machine learning theory to accomplish a particular objective using a particular structure. As such, using Q-Learning to solve MDPs built from medical time series data may prove to be an effective modeling strategy for binary classification of patient outcome. In the context of predicting mortality for Sepsis patients, the use of Q-Learning to solve an MDP may provide an accuracy benefit without requiring strict analysis of all co-founding factors used to construct the initial model.

2.4 Related Work

The inspiration for this paper is based on the 2019 "*Early Prediction of Sepsis from Clinical Data ... Computing in Cardiology Challenge*" provided by PhysioNet [27]. This challenge provided entrants with a database of various patient data, including whether or not a patient had Sepsis or not. The goal was to predict if a patient had Sepsis, 6 hours before the diagnosis from an actual doctor. During the judging phase, there was a much larger database of more patient data and diagnoses that the entrant's model was run against. The model gained points for every successful early diagnosis and lost varying amounts of points for late or incorrect diagnoses [51]. Many of these contestants utilized a form of Machine Learning, including the winning team which used a "...signature-based regression model" that took advantage of the techniques of both "...supervised and unsupervised machine learning models [52]."

The other central inspiration for this paper, also mentioned already, was research presented in a Nature article conducted by Komorowski et al.. The approach used by this team was an implementation of a Stock Q-Learning algorithm as available in authors' GitHub repository [26][1], making use of a custom evaluation metric. The AI clinician was tasked with a similar objective as the PsyioNet challenge models: provide an optimal treatment strategy for a given Sepsis patient.

[1] `https://github.com/matthieukomorowski/AI_Clinician`

Provided patient data, the AI clinician would supply an optimal treatment regiment by "...[extracting] implicit knowledge from an amount of patient data that exceeds by many-fold the life-time experience of human clinicians" in order to recommend a treatment [26]. Although the intent of this particular study focuses on treatment as opposed to diagnosis, the approach to the problem is similar.

Raghu et al. tackled the Sepsis treatment problem using what is known as "Double Deep Q-Learning" [53]. Their approach attempts to minimize the loss between the output of their model, and the target at every state. The "double" portion of this kind of network comes from the main Q-Learning network sending output Q-Values using a "feed-forward" pass, instead of having these values being determined directly from the target network. This kind of Q-Learning also makes use of what is called a "Dueling Q Network", which goes against the traditional model of each state being assigned a Q-Value for each given action. The Dueling Q Network splits the action-value pair into an additional parameter called "advantage", which determines the quality of the best action at a given state. In terms of policy choice, the researchers opted not to prescribe vasopressors (medicine that makes blood pressure go up) unless the patient had a very high SOFA score. The researcher's model ended up performing very well at diagnosing and treating patients with medium-level SOFA scores(5 - 15), but there was a significant increase in mortality rates for patients with very high SOFA scores($>$15). The researchers concluded that the model could be recommended for mid-level SOFA scores, but not for high-level SOFA scores due to lack of data.

In another reinforcement learning paper by Raghu et al., they were able to reduce patient mortality from Sepsis by up to 3.6% from the baseline mortality of 13.7% [54]. This model makes use of the previous hybridization of the "Double Deep Q-Learning Network" and the "Dueling Q Network" mentioned previously, with the addition of a new optimal policy technique. The "Doubly Robust Off-policy Value Evaluation" created by Jiang and Li [55] is mentioned here as the means of gathering a given Q-Value at a particular point. This paper also draws comparisons between two of the authors' created techniques: the "Dueling Double-Deep Q Network (DDQN)" and the "Sparse Autoencoder Dueling DDQN". In their experiment, both models developed good practices for prescribing vasopressors over IV fluids, but the autoencoder variant was much more accurate at

prescribing IV fluids (better than the human physician).

Peng et al. follow in the footsteps of the Raghu et al. group with another attempt using the DDQN method. The difference in this experiment is the blend of the DDQN approach with what is described as the Kernel Reinforcement Learning approach (KRL) [56]. The KRL approach groups nearby “neighbor states" and makes a distributed actions based on the closest neighbor states. Peng et al. proceeds to engage in what they describe as a “Mixture of Experts (MOE)" approach where they blend features from both the DDQN approach and the KRL approach. Also, they focused on optimizing a select set of factors, including age, Elixhauser (a medical index), SOFA, FiO2, BUN (Blood Nitrogen), Glasgow Coma Scale, Albumin, trajectory length, and max distance from neighboring states. In the results, the MOE approach behaved closest to the KRL approach but was benefited by the higher dosage nature of the DDQN approach. The authors conclude that the MOE approach outperforms the individual experts, as well as the clinician policy, with a note that more testing would need to be conducted on larger data sets to prevent extreme responses in outlier cases in particular.

Petersen et al. attempted a different route bearing the same idea in using Deep Q-Learning [57]. The authors of this experiment generated an MDP that deals with various cytokine groups in the human body. Cytokines are crucial in cell-signaling and are a way of knowing if something is wrong in the human body. The research group built their Deep Q-Learning model to treat areas where cytokine groups were producing active signals. Rewards were given to the model for a healed patient, and punishments for patients that died. The model was able to obtain 0% mortality on the data set it was trained on, and achieves 0.8% mortality on 500 randomly selected “patient parameterizations" that had baseline mortality values between 1% and 99%. The authors note that in this experiment, the model medicates cytokines for which no known drugs are known to be capable of doing, likely allowing for such high output values.

Tsoukalas et al. chose to solve a Partially Observable Markov Decision Process (POMDP) using an algorithm known as Perseus to build a Clinical Decision Support System (CDSS) for Sepsis [58]. The purpose of a CDSS is to give a clinician an assistant tool to recommend actions based on patients’ conditions, which are determined by the Perseus solution to the created POMPD. When

following the optimal policy, the resultant policy enabled 25.9% of patients to transition to a better state 90% of the time, while 33.7% of patients transition to a worse state 90% of the time. This is an improvement to the non-policy cases which were 12.9% and 51.2% respectively.

Hou et al. developed three machine learning models to predict mortality 30-day mortality in patients. Their investigation included the following: a logical regression model, SAPS-II score prediction model, and an XGboost model, all constructed in R [59]. Their dataset was comprised of 80.5% patients who lived, and 19.5% patients who died, all of whom met the Sepsis-3 definition of Sepsis. In addition, all of the patient data was extracted from the MIMIC-III dataset, previously mentioned and used in this research. Out of all the attempted models, the XGboost model was able to achieve the best results, achieving a 85.7% accuracy rate over their testing sets, making use of AUC techniques to compare result metrics.

Kong et al. constructed a model to predict in-hospital mortality of sepsis patients in the ICU, with the express purpose of determining optimal decisions based on a patient's condition severity [60]. They made use of 86 distinct predictor variables provided by the MIMIC-III dataset for their model input. In terms of modeling, they used four different types of models: *least absolute shrinkage and selection operator* (LASSO), *random forest* (RF), *gradient boosting machine* (GBM) and traditional logistic regression (LR) to predict. As for evaluation, they made use of the *simplified acute physiology score* (SAPS II) using five distinct metrics: Sensitivity, Specificity, Calibration Plot, and AUROC (*Area Under the Receiver Operating Characteristic Curve*). Out of all the models, GBM performed most accurately at 84.5%.

RESEARCH GROUP	***MODEL TYPE***	***RESULTS***
PhysioNet Winner [61]	Signature-Based Regression Model, Mix of Supervised + Unsupervised (Diagnosis Model)	Evaluated using a custom evaluation score that outperformed all other competitors based on ability to predict Sepsis.
Komoroski et al. AI Clinician [26]	Standard Q-Learning with Custom Evaluation Metrics (Intervention Model)	Evaluated using WIS offpolicy evaluation metric and performed better at intervention than the clinician policy 66.4% of the time.
Raghu et al. [53]	Dueling Double Deep Q-Learning (Intervention Model)	Specifies observed mortality using AI policy as graphed continuous data. AI outperforms the clinician model at lower SOFA scores, but not at higher SOFA scores.
Raghu et al. Optimization [54]	DDQN and Sparse Autoencoder DDQN (Intervention Model)	When evaluated on past ICU patient data, the model was found to reduce in-hospital patient mortality by up to 3.6% using the model's suggested interventions.
Peng et al. [56]	MOE (DDQN + KRL) (Intervention Model)	The MOE model outperforms all other models (DQN, Kernel, and Clinician) under the WDR estimator metric.
Petersen et al. [57]	Stock Deep Q-Learning on Cytokines (Intervention Model)	For patients on Multi-Cytokine medication, the model performs at 0% mortality on the training set, and 0.8% mortality over 500 randomly selected Sepsis patients.
Tsoukalas et al. [58]	POMDP + Perseus Algorithm (Intervention)	Under the optimal policy, 25.9 % of patients had 90% of their states transition as better transitions, 33.7% had 90% of their transition to worse states, compared to the clinician policy, which were 12.9% and 51.2% respectively.
Hou et al. [59]	XGBoost Model (Diagnosis Model)	XGBoost achieved 85.7% accuracy on a subset of MIMIC-III consisting of 80.5% living patients, and 19.5% dead, with a total of 4559 patients.
Kong et al. [60]	Gradient Boosting Machine (Diagnosis Model)	The GBM model variants used 16,688 MIMIC-III patients, 82.3% of which died, and 17.7% lived, achieving an average AUROC score of 0.845.

Table 2.5: A Summary of all Related Work

Chapter 3

Methods

3.1 Introductory Information Overview

Prior to going into detail on the methodology, an overview of the factors integral to the development of this project provides an understanding of how the research was conducted. Also, the information provided can be used to provide clarity for future research efforts, outlines a source of maintainability and reproducibility, and future-proofs the work by exposing any errata concerning language or hardware dependency errors (*e.g.* bugs in the Python language, or problematic driver interaction) which may be revealed in the future.

3.1.1 Chosen Technology Stack

All development for this research was conducted on JuptyerLab, installed through the Anaconda distribution environment. Versions for which are displayed in Table 3.1. As for programming language selection, the entire program pipeline was written and tested in Python Version *3.7.6* within the JupyterLab IDE. Several parts of the code that were used for the initial data processing and filtering was developed in MATLAB. All code that ran preliminary data processing was done using MATLAB version *R2020a Update 3 (9.8.0.1396136)* as well as MATLAB Kernel for Jupyter, version *0.16.11*. All development was conducted on the *Windows 10* operating system. Finally, for version control, Git Version: *2.16.1.windows.3* was used with a remote location on GitHub, available here[1]. Additional pieces of technology were used in the dataset construction phase, but these will be discussed at length in a subsequent section.

[1] `https://github.com/newtnewtnewt/FUTON_Research`

PROGRAM	VERSION
jupyter core	4.6.1
jupyter-notebook	6.0.3
qtconsole	4.6.0
ipython	7.12.0
ipykernel	5.1.4
jupyter client	5.3.4
jupyter lab	1.2.6
nbconvert	5.6.1
ipywidgets	7.5.1
nbformat	5.0.4
traitlets	4.3.3

Table 3.1: JupyterLab Version Information

3.1.2 Data Sourcing

PhysioNet, through the MIT Lab for Computational Physiology, provides one of the largest publicly accessible intensive care databases in the form of *MIMIC-III*. MIMIC-III covers over a decade of medical records, and offers free and open access to anyone who has obtained "CITI 'Data or Specimens Only Research"' class completion.[2] In particular, the data covers over a decade of patient medical data, chart information, vitals data, and diagnosis information [62]. Due to the work of researchers who have come before this point, MIMIC-III has been made much more manageable for coming up with a Sepsis specific solution. Komoroski et al., built a full MATLAB script[3] that extracts patient information from the MIMIC-III data set. The result is a significantly more refined data set containing patients diagnosed with some form of Sepsis, as well as their vitals. More details on this are available in the subsequent data processing section. Komoroski

[2] MIMIC-III can be found here: `https://mimic.physionet.org/`

[3] Source Code can be found here: `https://github.com/matthieukomorowski/AI_Clinician/blob/master/AIClinician_mimic3_dataset_160219.m`

et al. also provides a script to extract all the Sepsis-III definition patients and clean their values that are out of range or not correct in some way. This data set takes the final form of a single file, *'patient_data.csv'*, which is not available publicly due to data access restrictions. Instructions on reconstructing the dataset are available through PhysioNet, as well as in the research repository for this project.

3.1.3 Hardware Overview

All development on this project was conducted on a single machine, and all optimization runs were conducted on a separate collection of machines: a high performance computing cluster. Specifications for which are available in Table 3.2.

DEVELOPMENT PC	***NAME***
CPU	Intel (R) Core (TM) i7-4770K CPU @ 3.50GHz - 4 Cores
GPU	NVIDIA GeForce GTX 1070 Ti
MEMORY	16 GB
REDHAWK CLUSTER COMPUTE NODE	***NAME***
CPU	Intel(R) Xeon(R) Gold 6126 CPU @ 2.60GHz - 24 Cores
GPU	ASPEED Technology, Inc. ASPEED Graphics Family (rev 41)
MEMORY	96 GB

Table 3.2: Hardware Specifications for Development PC and Computing Cluster

As a final note, for local storage, all executables, data, and programs were kept on a *Seagate One Touch SSD 512 GB* for ease of transfer and accessibility.

3.2 Data Processing and Filtering

Before building a machine learning model using any technique, it is essential to properly process and clean that data. Proper data processing leaves the eventual model to be less susceptible to rogue

environmental factors, and it also enables the model to remain closer to real world circumstances. During the data collection process, it is often that values will be forgotten, missing, or inputted incorrectly, and this stage of Machine Learning seeks to remedy such problems. Since the scripting for this process was done entirely by the Komoroski et al. research group [26], the explanation of what was done to pre-process the data is provided in Appendix *A.1*. Since the work present here directly builds on the process established and validated by Komoroski et al., the reader is invited to consult [26] for details on their data processing.

3.3 Model Design and Pipeline Construction

The *FOOTON* model, a binary classification model named due to its use of qSOFA and SOFA scoring, is the artifact produced by the efforts of this research. Ultimately, the *FOOTON* model constructs a model that takes real world ICU data from the MIMIC-III dataset as input, and it outputs four variants of a model with the ability to predict whether or not a patient will live or die given all known information provided. The intricate details of how that is accomplished are documented in the subsequent sections.

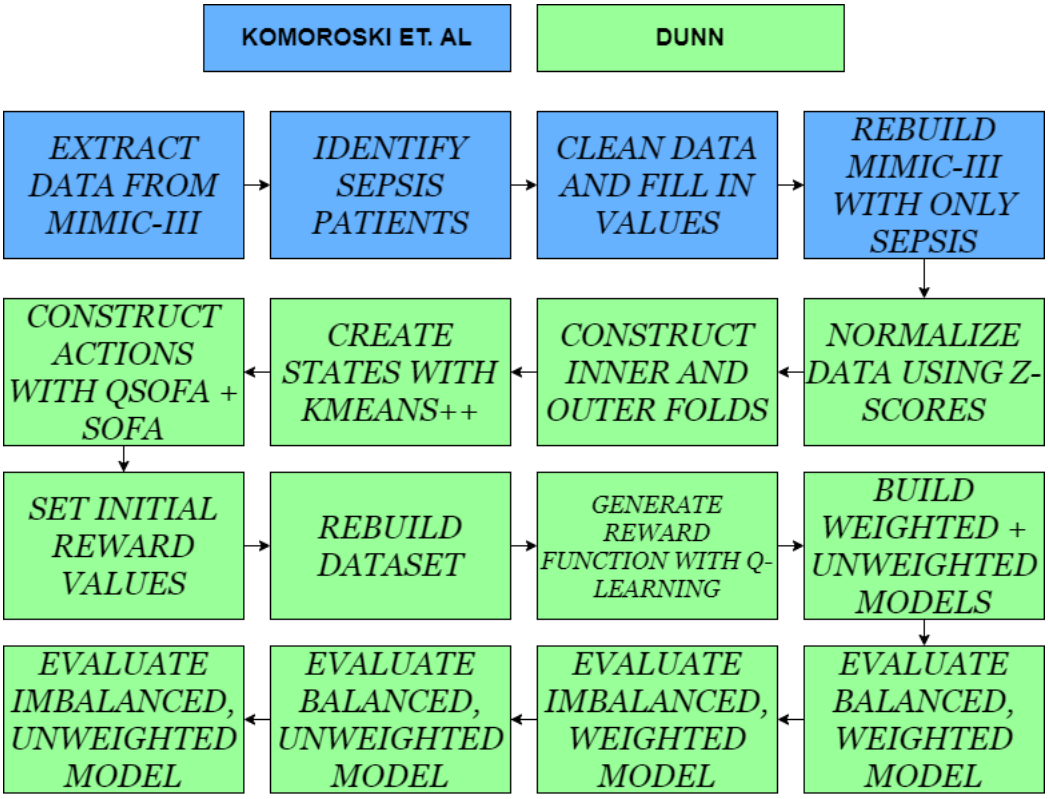


Figure 3.1: A High Level Overview of the Methods

3.3.1 Normalize the Data

In addition to the thorough data cleaning and processing that occurred during the pre-processing phase in *A.1*, it is necessary to partition data into groups to effectively construct and test models. For the sake of this experiment, all data that is used to construct a given model will be referred to as the "training set", and all data that is used to test the performance of that model will be referred to as the "testing set" which is the standard convention in practice and the literature.

Initially, the fully pre-processed data represented in the file "patient_data.csv" is loaded into a Pandas' DataFrame. From the initial dataset, three sets of columns are chosen: binary columns, normal columns, and logarithmic columns. These sets are named based on the type of data they represent. If these values were used in a different form of modeling with no summary statistics for identification, these three categories would matter to a much greater degree in the overall model. However, due to the usage of Z-Scores and the later usage of KMeans++ clustering in Section 3.3.3,

the difference between these column types is in name only. Nevertheless, a table of the final chosen factors and their respective data classifications are present for convenience in Table A.3.

With some modeling techniques, it is encouraged to use columnar data values in the format they have been provided. However, when using a clustering technique (such as *KMeans++*, used for this project and discussed later in Section 3.3.3), it is vital to standardize the data, as is done by calculating Z-Scores. Calculating and using Z-Scores across the entire data set effectively determines distances from an average present in that column, but more importantly, it prevents columns with very large differences in values (*e.g.* Weight) from having more of an impact on the clustering process than columns with small differences in values (*e.g. Sex*). At this stage, all values are normalized to the range -3 to 3 calculated on a per-column basis.

3.3.2 Split Data Into Train and Test Sets

After the dataset was modified entirely to Z-Score format, there was still the necessity to split the data into training and testing sets, as well as to have some validation of the stability of the model across sets. To resolve this issue, a 10-Fold Stratified Nested Cross-Validation was used, shown in Figure 3.2. In standard k-fold cross-validation, the total data set is divided into 10 training sets and 10 testing sets. The number 10 is used as a "golden" standard for k-fold cross validation, as its performance in various experimental studies has produced results with lower bias and lower variance as opposed to other values of k [63]. The 10 training sets are used to build models, which are then evaluated on their 10 testing set complements (*e.g.* testing set 0 is evaluated on the model built from training set 0, so on and so forth). The "folds" in k-fold validation are each of the training and testing set pairs used in the experiment (10 folds yields 10 training and 10 testing sets). In a proper 10-fold validation like is performed here, each training set is constructed with approximately 90% of the patients, with 10% of the patients being left out to build the testing set[4]. Two versions of the data were created at this stage, a balanced training set and an imbalanced training set, discussed later.

Discussing the general technique, cross validation has been shown in experimental studies to

[4] https://machinelearningmastery.com/k-fold-cross-validation/

produce more accurate and less biased results compared to alternatives techniques, including the simple case of allocating a separate validation data set [64]. As mentioned previously, this is not a standard cross-validation, this is both *stratified* and *nested* as well. *Stratification* of the data occurs when a percentage of samples for each class is represented in the training sets. In the case of this research experiment, a stratified dataset is going to ensure that all 10 training sets have patients involved with high, medium, and low body weight, high, medium, and low blood pressure, etc. . . .

Since the input data for this model is time-series data (patient's vitals over time), and one patient represents many rows of data, it is necessary to profile a single patient by calculating some summary statistics of their entire time in ICU. This is necessary because otherwise the stratification process would take individual rows of data and generate datasets that did not include all of a given patent's data values. To prevent this, a separate pandas DataFrame is created as an intermediate data set built using summary statistics[5] for every given patient. The temporary DataFrame was fed into the StratifiedKFold function provided by the sklearn library[6]. StratifiedKFold produces sets of patient ids that were used to extract all the patient data from the original sets for use in building the actual models.

The last piece of this process, the fact that this k-fold process is *nested*, indicates that there are two sets of folds generated. The first set, called the outer folds, are the original set that is used for the training and testing of the model. The second set, called the inner folds, is used to optimize the hyper-parameters for each training and testing set. Hyper-parameters are discussed to completion in the later Q-Learning section, but to conclude this section, the inner folds are used to optimize hyper-parameters with datasets that will not over-fit the model or generate a large amount of additional bias or noise [65]. In summary, this research uses the 10-fold Stratified Nested Cross-Validation to train and test on datasets that produce models that with stable hyper-parameters that are low in data bias and low in variance.

[5] Mean, Min, Max, Q1, Q3 were calculated for each column

[6] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html`

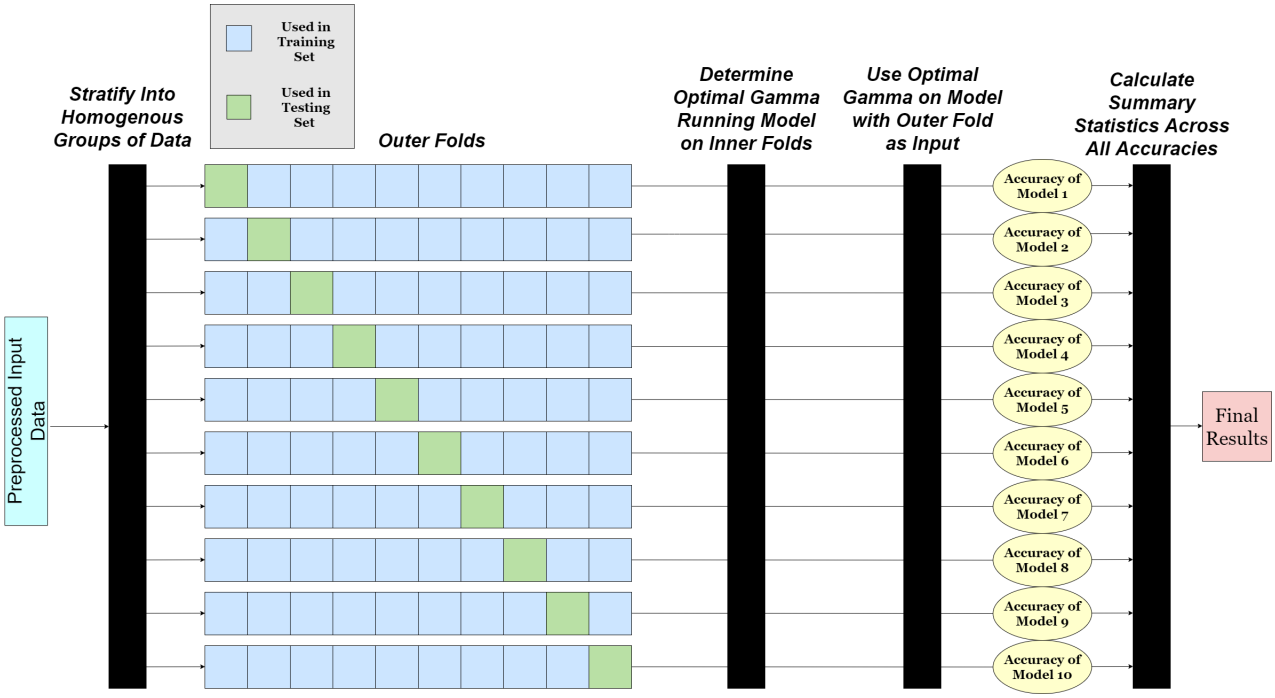


Figure 3.2: Stratified, Nested 10-Fold Cross Validation

3.3.3 Create States Using KMeans++ Clustering

The first step to creating a model with Q-Learning is constructing an MDP from the data, as discussed in the background section. For simplicity, instead of following the path of all 10 outer folds and all 10 inner folds, this explanation is going to cover a single fold: one training set and one testing set.

Recall, all MDPs require states, actions, and a reward function to exist. For medical time series data, specifically patient data, a patient goes from one "state" to another when any of their vitals change, which happens at every single timestamp (age is the easiest factor to consider this for, every second that passes changes a patient's age). However, generating a state for every single patient timestamp for all patients is not computationally or practically feasible. Doing something like this would result in exponentially larger model build times for each patient added to the system, and it would over-fit the model to require very fine-tuned inputs to receive any results. Instead of this, the KMeans++ algorithm is used to group rows of like patient data into states.

The KMeans++ algorithm provided by sklearn[7] converts a dataset into K smaller set of nodes called *clusters*, using Euclidean distances over many iterations. The value for K can be selected using several techniques; however, this project used a more cutting edge technique for selecting the optimal value for K known as the "curvature method". Developed by Zhang et al. [66], the curvature method takes variance output produced by the KMeans++ algorithm between a specified range (for this experiment it was between 1 and 100) as all the desired values of k. Along all of these data points a curve is plotted, as shown in the reduced example in Figure 3.3.

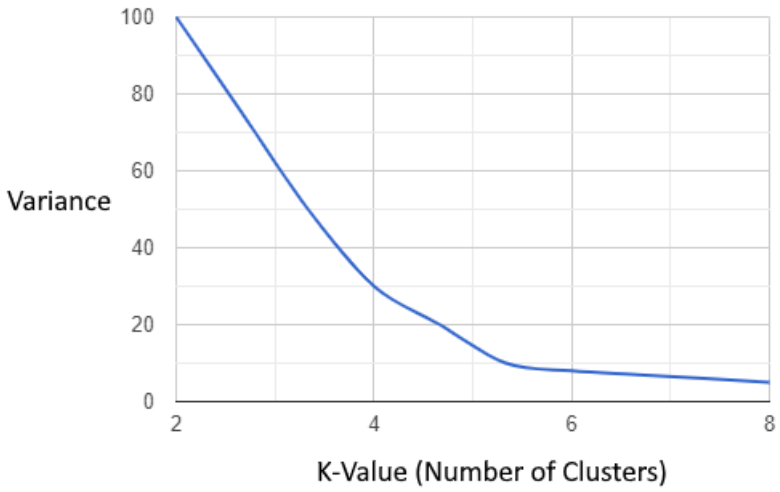


Figure 3.3: A sample Curvature plot between K=2 and K=8 [66]

Since graphics are based on the scale of the x and y axes, determining maximal curvature at this point would be susceptible to error. Any changes to either of the axes could produce a better K-value simply by modifying the scale and keeping the data the same, the curvature method attempts to avoid this. Instead, for all values of K, the curvature method runs through a series of equations making use of single parameter alpha, over a fixed range. The k-values are plotted against the equation results for each alpha, and whichever k-value yields the maximal curve is deemed to be the optimal value. In this experiment, *K=74* ended up being the optimal K-value, and a simplified

[7] `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html`

example of this process is shown in Figure 3.4

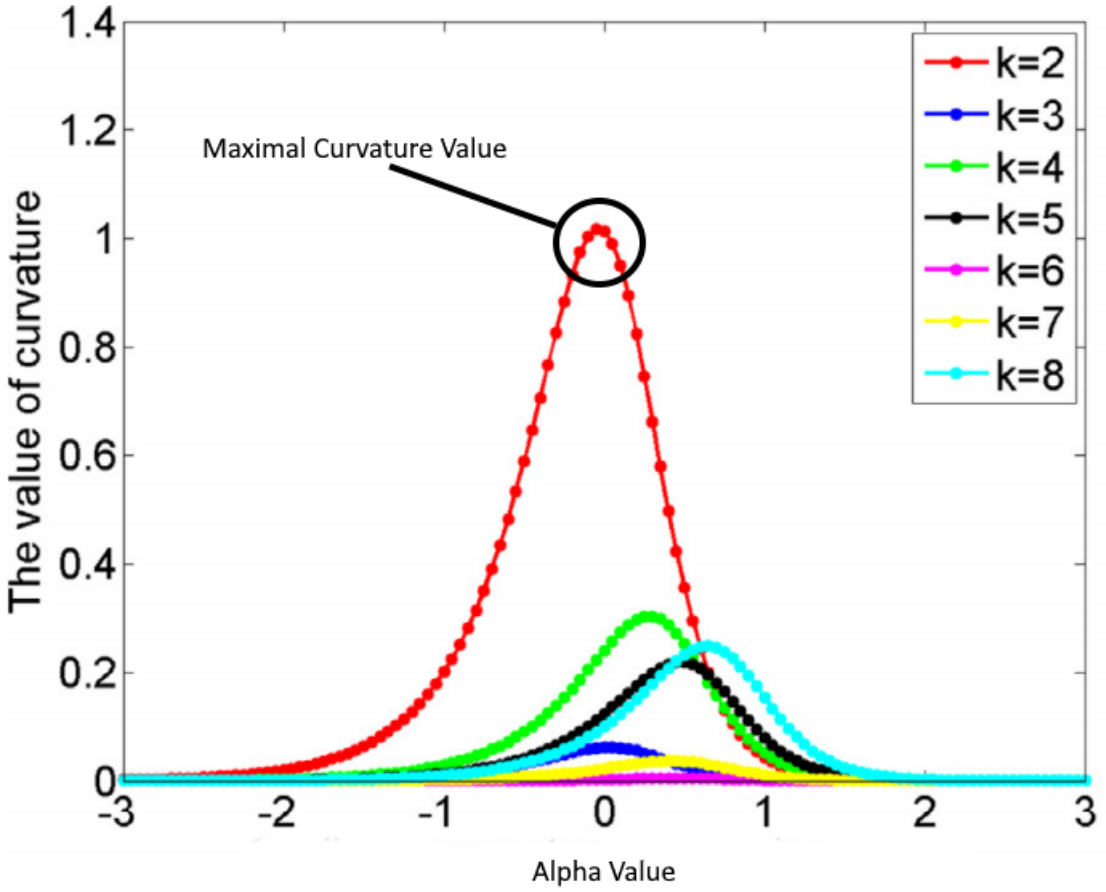


Figure 3.4: Determining optimal K value between K=2 and K=8 [66]

The Z-Score version of the patient dataset was fed into the KMeans++ algorithm, running for its default value of 10,000 iterations, at its default value of 32 runs per update iteration. The output was a DataFrame that mapped every row of data to 1 of the 74 states produced in the K-Means algorithm.

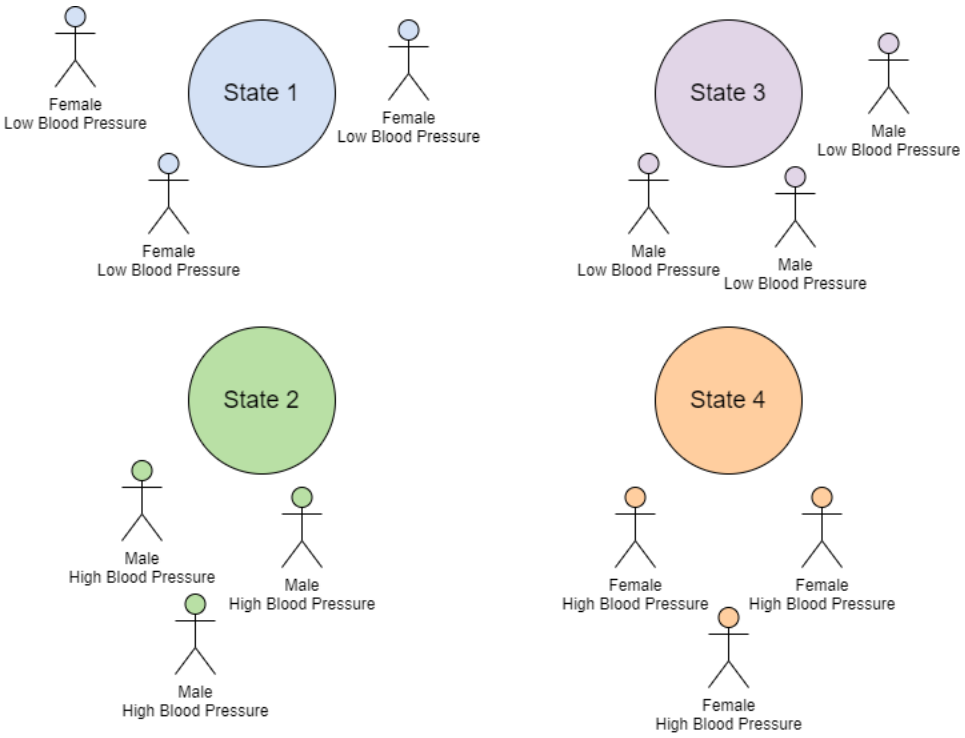


Figure 3.5: An example of Clustering Patients according to their Data

3.3.4 Build Actions using qSOFA and SOFA scores

With the full set of states constructed from the data, 1 of the 3 necessary components of the MDP has been designed. The next component, actions, are what enable the states to connect in meaningful ways.

One of the prominent focuses of this model was to experiment on the use of qSOFA and SOFA over patients' stay in the ICU. This model groups ranges of qSOFA and SOFA values into *20* unique actions, and assigns each patient one of these actions for each time-step in their data, depending on their qSOFA and SOFA scores at that given stage. Recall from the introduction, SOFA scores range from 0 - 23, and qSOFA Scores range from 0 - 3. Splitting each of these into collections of ranges and then combining them into single actions is shown in Table 3.3.

Note that Action 19 is listed for completeness, but due to the Glascow Coma Score being involved in both SOFA and qSOFA score calculation, it is impossible to have a SOFA score of 20 and not have a qSOFA score of at least 1. Following the creation of the action matrix, a new

	SOFA				
qSOFA	0-1	2-7	8-13	14-19	20-23
0	0	4	8	12	**19***
1	1	5	9	13	16
2	2	6	10	14	17
3	3	7	11	15	18

Table 3.3: Matrix for Determining Action Value
* Action 19 is not Possible

DataFrame is created, similarly to the state DataFrame, which runs through every time-step of patient data and assigns one of the actions shown in the action matrix table to the row based on the qSOFA and SOFA values at that point.

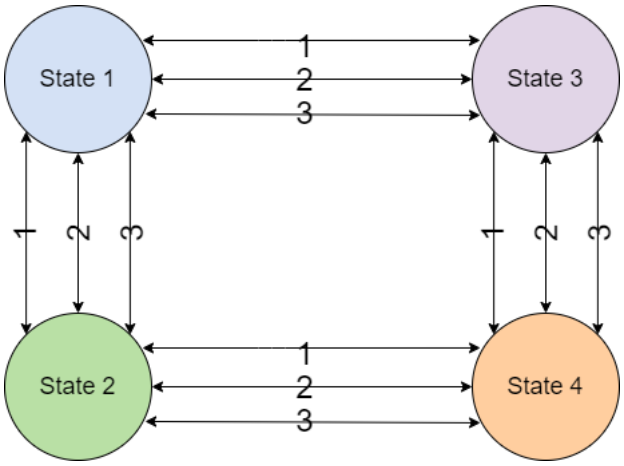


Figure 3.6: A Trivial Example of Action Generation, with 3 Actions and 4 States

3.3.5 Implementing Initial Rewards

At this stage, 2 of the 3 requirements for building the model have been fulfilled, leaving only the initial rewards left. Prior to using Q-Learning, the state, actions, and rewards need to be formatted in a way that makes sense for the progression of the Q-Learning algorithm over a series of time-steps.

For the proper storage of reward values, two additional terminal states were added to the initial set of 74, one representing life and one representing death. A matrix of size 20 x 76 was created

to save rewards representing a transition to any state and action combination. The initial values for all rewards was set to 0 except the additional 2 states, 74 and 75 (counting starting at 0), which represented life and death. All actions that led to the state of life were default set to a reward of +100 and all actions that led to the state of death were default set to a penalty of -100. For example, the value m^{2}_{34} in the matrix would indicate the reward value assigned when taking the 2nd action from the 34th state.

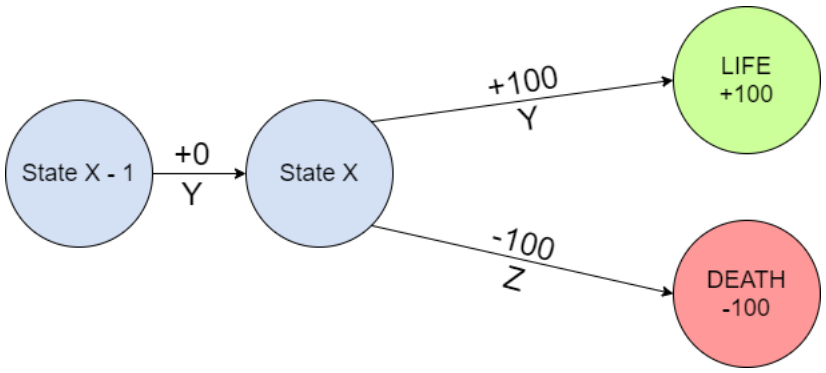


Figure 3.7: An Example of Initial Rewards for an Unweighted Model

Additionally at this phase, another matrix was created similar in structure to the reward matrix. For every state, this matrix calculated the probability of taking a given action with respect to every other action. This was a simple calculation done based on the data:

$$\frac{\text{frequency of action A occurring from state S}}{\text{frequency of all actions A occurring from state S}} \tag{3.1}$$

This is used for the weighted variant of the model, described later, that factors in action probabilities in calculating the reward at every given point.

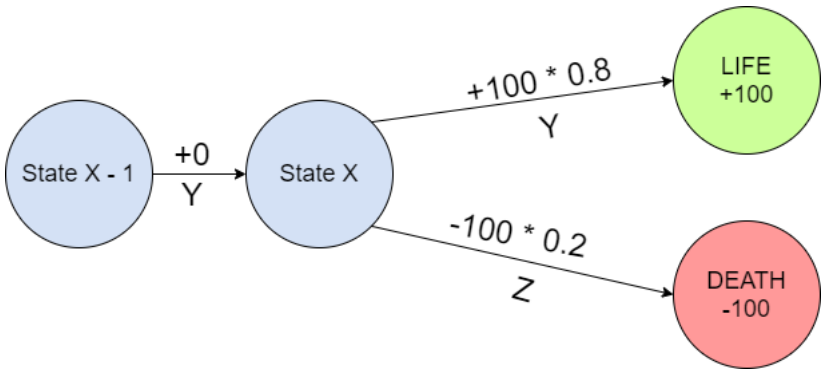


Figure 3.8: An Example of Initial Rewards for a Weighted Model

3.3.6 Building the Interim Dataset

With the state, actions, and initial rewards established, a new interim dataset can be reconstructed to input into the Q-Learning algorithm, before the creation of the final evaluation set.

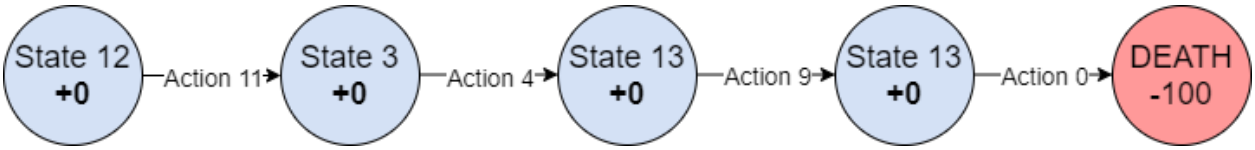
For every patient in the training and testing sets, each time-step is converted into a row of 6 column values:

1. Training Bloc (Time-Step)
2. State
3. Action To Take
4. Reward
5. Mortality Status After 90 Days
6. Patient ID

In addition to the normal time-steps, a final time-step is created for every patient at the end of their data to account for their terminal state (life or death). At this step, the state is set to either life or death, the action to take is set to -1 (No more actions can be taken from the last state), the reward is set to either -100 or +100 depending on life or death, and the rest are as normal. The terminal state enables proper reward distribution as discussed in the Q-Learning phase of the experiment, discussed in a subsequent section. These modifications are the last that occur to generate the DataFrame that is input to the Q-Learning algorithm, used to generate finalized accurate rewards.

Also, the modifications are made to the testing set, to be used after the Q-Learning process takes place on the training set.

(a) A Sample of a Living Patient Prior to Q-Learning



(b) A Sample of a Dead Patient Prior to Q-Learning

Figure 3.9: An Initial Example of Living and Dead Patients in the Interim Dataset

3.3.7 Generate Reward Function Using Q-Learning

In order to make the model useful, the MDP needs to have reward values at each step that is reflective of the underlying data for a given state-action pairing. A modified version of Q-Learning is what is used to accomplish that task.

This version of Q-Learning is referred to as "modified" Q-Learning throughout the paper because it makes two modifications to the traditional Q-Learning algorithm.

1. It removes the "learning rate" hyper-parameter, alpha
2. It constructs the Q-Equation based on what actually happened in the data, not based on the "optimal action" to take at a state

The "learning rate" hyper-parameter was removed in favor of using an average across all runs, because it yielded better experimental results. In addition, choosing an optimal action given a state does not make sense for a reward function, given that this is a binary classification model and not an intervention model. All states and actions are predetermined and the reward function is supposed

to emulate the fate of the patient, not attempt to modify what occurred in any way. The process of generating the Q-Equation reward function from the data is shown in Figure 3.10.

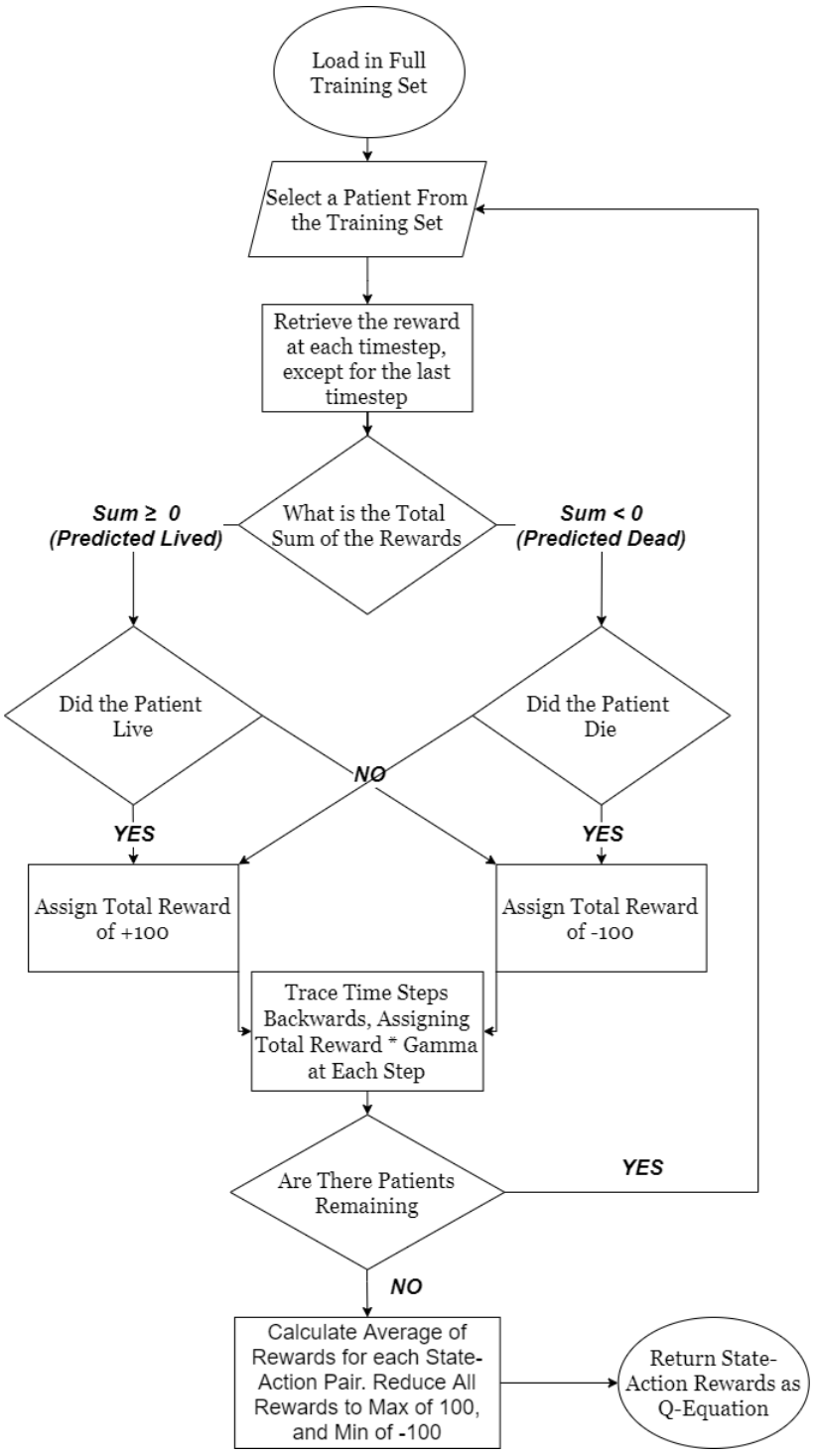


Figure 3.10: Algorithm for Generating the Q-Equation for all Rewards

After the Q-Learning process produces a Q-Equation of state-action reward values, the final step is to generate the evaluation set used to determine accuracy. The interim testing dataset constructed in the previous step is left alone, except for one modification. All the reward values are edited for each time-step based on the Q-Equation, generated from the training set. This finalizes the preparation for evaluation of the results.

(a) A Sample of a Living Patient After Q-Learning

(b) A Sample of a Dead Patient After Q-Learning

Figure 3.11: An Example of Living and Dead Patients in the Final Dataset, Ready for Evaluation

3.3.8 Evaluating the Test Sets

By finalizing the dataset into a format that uses real state, action, and reward values, the final stage of the pipeline can execute. The evaluation phase determines how well or poor each trained model variant does at predicting data from the testing sets it has not had any exposure to previously. The modified test set produced by the results of the Q-Learning algorithm contains all the correct reward values for each state-action value pairing. The evaluation for performance for the test set is done in a very simple manner that is similar to the Q-Learning reward assignment phase, shown in Figure 3.12.

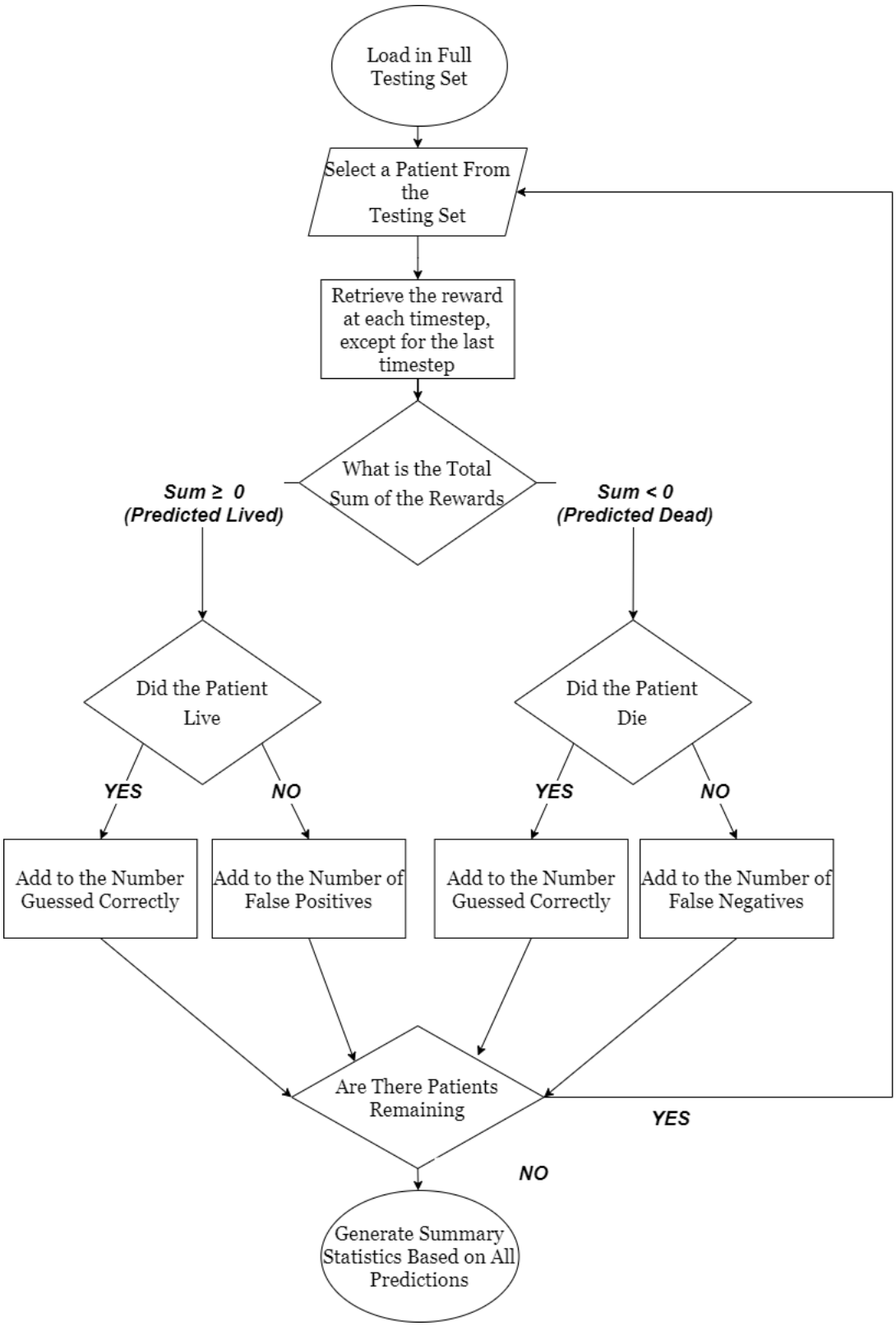


Figure 3.12: Algorithm for Evaluating Model Performance on the Testing Set

This very simple means is a way of determining accuracy, and false negative/positive rate for the model on the testing set. This is standard for most binary classification models. Additionally, for the sake of this study, displayed in Table 3.4.

- A *False Positive* indicates that a patient was predicted to live but actually died
- A *False Negative* indicates that a patient was predicted to die but actually lived

Two additional modifiers were used to test the final models: *weighted* or *unweighted*, and *balanced* or *imbalanced*. The balanced variant of the model chooses to balance the training set [67] [68] prior to the process of the Q-Learning, allowing only around 50% of the patients in that set to be patients who live, and 50% to be patients who die. In addition, the weighted variant of the model, after the Q-Learning process, applies a weight to each of the rewards based on the probability of that state-action path occurring. For example, if the reward produced for the state-action pair State 17, Action 12 by the Q-Learning algorithm was 45, and the probability of the action 12 occurring given that the state was 17 was 50%, the weighted variant model would assign the actual reward:

$$\mathbf{0.50 * 45 = 22.5} \tag{3.2}$$

The idea behind this is that potentially, if there is a sparse state-action pair with a highly negative or highly positive reward, it will not bias all test sets that pass through it with an extreme value. In the end, four final models were produced for comparison, shown in Table 3.5.

	The Patient was Predicted to Die	**The Patient was Predicted to Live**
The Patient Died	True Negative	False Positive
The Patient Lived	False Negative	True Positive

Table 3.4: The Confusion Matrix Layout

	The Training Set Is Balanced	**The Training Set is not Balanced**
The Rewards are Weighted Based on Probability of Occurrence	Balanced, Weighted Model	Imbalanced, Weighted Model
The Rewards are not Weighted Based on Probability of Occurrence	Balanced, Unweighted Model	Imbalanced, Unweighted Model

Table 3.5: The Final Four Model Types

3.4 Results

Model	*Imbalanced Weighted*	*Imbalanced Unweighted*	*Balanced Weighted*	*Balanced Unweighted*
Average Overall Accuracy (%)	77.978	**78.413**	72.204	70.946
Standard Dev. For Overall Accuracy (%)	0.343	0.399	1.321	1.152
Average Living Accuracy (%)	**99.086**	98.401	77.503	73.818
Standard Dev. For Living Accuracy (%)	0.281	0.229	1.400	1.284
Average Dead Accuracy (%)	9.552	13.614	55.029	**61.638**
Standard Dev. For Dead Accuracy (%)	1.449	1.560	2.261	2.104
Average Actual Living (%)	76.249	76.249	76.249	76.249
Standard Dev. For Actual Living (%)	$5.177e^{-5}$	$5.177e^{-5}$	$5.177e^{-5}$	$5.177e^{-5}$
Average Actual Dead (%)	23.575	23.575	23.575	23.575
Standard Dev. For Actual Dead (%)	$5.177e^{-5}$	$5.177e^{-5}$	$5.177e^{-5}$	$5.177e^{-5}$

Table 3.6: Summary Statistics for the Results of All Four Versions of the Model

N = 2144	**Predicted: DEAD**	**Predicted: LIVE**
Actual: DEAD	True Negative: 48	False Positive: 457
Actual: LIVE	False Negative: 14	True Positive: 1625

Table 3.7: Confusion Matrix for Imbalanced, Weighted Averages

N = 2145	**Predicted: DEAD**	**Predicted: LIVE**
Actual: DEAD	True Negative: 68	False Positive: 437
Actual: LIVE	False Negative: 26	True Positive: 1614

Table 3.8: Confusion Matrix for Imbalanced, Unweighted Averages

N = 2144	**Predicted: DEAD**	**Predicted: LIVE**
Actual: DEAD	True Negative: 278	False Positive: 227
Actual: LIVE	False Negative: 368	True Positive: 1271

Table 3.9: Confusion Matrix for Balanced, Weighted Averages

N = 2144	**Predicted: DEAD**	**Predicted: LIVE**
Actual: DEAD	True Negative: 311	False Positive: 194
Actual: LIVE	False Negative: 429	True Positive: 1210

Table 3.10: Confusion Matrix for Balanced, Unweighted Averages

3.4.1 Discussion of Results

Of all four models, the Imbalanced, Unweighted Model performed the best in overall accuracy, yielding 78.978% accuracy in predicting patient outcome correctly. As for the ability to predict living and dead patients correctly, the Imbalanced, Weighted model had the highest accuracy for predicting living patients at 99.086%, and the Balanced, Unweighted model had the highest accuracy for predicting dead patients correctly at 61.638%. In the case of the imbalanced model, it is important to note that it is likely over-fit, yielding better results for the living due to a higher input of living patients.

The confusion matrices present in Table 3.7 to Table 3.10 demonstrate how the balanced and imbalanced models differ in their choice of accuracy[8]. The imbalanced models heavily favor predicting patients to live due to the abundance in the data (76% of the patients in the imbalanced training set live), which biases their models towards living patients. The balanced models, like the name suggests, provide a more balanced approach, predicting a much higher number of the dead patients correctly.

In practice, a hospital would benefit from a balanced model, specifically the weighted variant. If the patient is treated as if they are going to die, as is the case for the balanced, weighted variant with a much lower false positive rate, this is likely to lead to more attentive treatment than if the model expects the patient to live. Hospitals are going to be more concerned with patients with a risk of dying, as opposed to those who are likely going to live.

It is also important to note that if all models are compared against a perfectly optimistic model (a constant model that predicts every patient will live), the models in question must perform better than 76.249% overall accuracy, as this is the average percentage of patients who live in each of the testing sets. The imbalanced, weighted model performs 1.729% better than the overly optimistic variant, and the imbalanced, unweighted model performs 2.164% better than the overly optimistic variant in overall accuracy. The balanced, weighted model performs 4.045% worse than the overly optimistic variant, and the balanced, unweighted performs 5.303% worse than the overly optimistic variant in overall accuracy. However, the overly optimistic model predicts dead patients at a rate of

[8] The value of N in the matrix for the imbalanced, unweighted model is off by 1 due to using consistent rounding on the averages.

0.000%, as opposed to the 9.552%, 13.614%, 55.029%, and 61.638% dead accuracy of the other four, listed in their respective order.

3.4.2 Alternate Approaches Explained

In place of using raw Z-Score values prior to clustering, Principal Component Analysis (PCA) was tested. In short, PCA is used when there is little intuition behind the factors being used to build the resultant models. It transforms the columns of the data such that the smaller number of transformed columns suffices to represent a certain variance in the data (Z-Scores in this case). No extensive hyper parameter testing was conducted, but short tests were run using PCA including $\geq$ 80% of the variance from the original dataset. In all cases, the performance was less than the non-PCA counterpart.

The initial version of the model included the growth rate hyper-parameter, alpha, which was removed due to suffering severe bias to the way in which the data was inputted. It also produced accuracy values that were less than the averaged reward version described previously. Modifications to the reward function itself were also tried, adding rewards to certain state-action pairs or observed patterns. None of these were exhaustively tested. Preliminary tests consisting of runs on a few of the test cases, without hyper-parameter optimization, on all modifications produced a much wider variance in model performance output. Trimming down factors into only the ones that were directly related to Sepsis was another consideration, but leaving information out of the model tended to make it weaker for all versions.

In addition, instead of using the curvature method to determine the value of K for the K-Means clustering, sparser numbers of clusters were tried: 5, 10, 15 and 20. In all cases, for both balanced and imbalanced training sets, the patients were predicted to be living at a much higher rate than normal, yielding a lower overall accuracy. As a final consideration, the penultimate state-action combinations were looked at for each patient, but there were no significant enough observable patterns to modify the model or reward model in any meaningful way.

Chapter 4

Conclusion

The efforts of this study have determined that is feasible to use a modified version of Q-Learning to predict mortality in Sepsis patients. Specifically, the FOOTON models produced by this study are able to perform better in some criteria than a “proportional random" or “overly optimistic model". The unweighted and weighted imbalanced models perform better than the random/overly optimistic models in overall accuracy, and the weighted and unweighted balanced models outperform all other models in the dead patient prediction rate. The models were tested on data sets diverse in patient demographics, across a fairly large test set, compared to other forms of mortality prediction models, which specify a particular demographic on a smaller, more specific test set. The FOOTON model in its current state is likely not ready to be implemented into medical decision assistance technology, but perhaps, with future research and developments in the realms of machine learning and Sepsis research, this model can prove to be a contender in helping real physicians make life and death decisions to maximize saving patients' lives.

4.1 Future Work

For future research, there are still a number of considerations that can be tackled to construct a potentially better model. One idea is to use one of the other metrics for mortality provided by the initial dataset instead of living status after 90 days. Either using the mortality statistic after 48 hours, or if there would be the possibility of obtaining mortality status in ICU, that may pose a more effective model. Another factor that may be useful to examine is the length and duration of antibiotic and vasopressor administration, both of which are used in treating Sepsis. Also, no

additional weight was provided in the case that a patient was revisiting the ICU, which may have been indicative of a higher chance of death.

Recall that this study used a window of 24 hours before to 48 hours after a patient had been infected with Sepsis. It is possible that trying a shorter or larger window may yield much different results. As a final note, separating patient data based on known demographic values prior to clustering (based on presence of sepsis shock, age demographics, etc...) and generate models for each may yield higher overall accuracy. If there was a metric gathered to determine the time-stamp at which a patient began receiving treatment, this may be a useful factor to add a weighted reward to in a given patient's outcome evaluation.

As a final consideration, it is possible that using an approach based on Q-Learning may not be the best technique to use for this particular problem. Specifically, other attempts at mortality prediction for ICU patients (not dealing with Sepsis directly) have been made using alternative modeling techniques, in addition to those listed in the related works. Awad et al. made use of a time series analysis model to predict hospital mortality in ICU patients [69]. Sinuff et al. constructed a model using a scoring system that was evaluated on a patient-by-patient basis to predict ICU mortality outcome [70]. Lipshutz et al. made use of modeling built from logistic multivariate regression in order to predict ICU patient mortality in the ICU [71].

Appendix A

Additional Tables, Figures, and Information

A.1 Komoroski et. al's Data Processing

A.1.1 Extract Data From MIMIC-III

MIMIC-III is a public dataset, but because of medical and data integrity restrictions, the dataset is stored behind several online classes that are required to be taken on the topic of data integrity. For this purpose as well, it was not possible to upload these to GitHub. To get around this issue, the authors provide a section of their web page that gives access to all the CSV files which is only accessible to users who have passed the required courses and who have been granted access [1]. The data for MIMIC-III is divided among dozens of compressed CSV files, ranging from as low as several KB, to several GB for the largest dataset.

This stage of the data is distributed, mangled, and obfuscated beyond all readability. Fortunately, the research group for the AI Clinician, Komoroski et. al, provides a series of scripts to make data extraction for the mangled dataset streamlined[2] [26]. First, the full MIMIC-III database had to be constructed in PostgreSQL using the scripts provided by the MIMIC-III development team [3]. After all the tables were constructed in the local database, the Komoroski et. al Jupyter scripts were run which pulled the following data out of the database:

- Blood/Urine/CSF/Sputum Cultures
- Antibiotics Administration

[1] https://physionet.org/content/mimiciii/1.4/

[2] https://github.com/matthieukomorowski/AI_Clinician/blob/master/AIClinician_Data_extract_MIMIC3_140219.ipynb

[3] https://mimic.physionet.org/tutorials/install-mimic-locally-ubuntu/

- Demographic Data
- Chart Event Information
- Lab Event Information
- Output Event Information
- Microbiology Event Information
- Fluid Intake
- Vasopressor Intake
- Mechanical Ventilator Presence

All of the above were placed into separate CSV files containing over *1,000,000* rows each. At this stage in the data processing, there are a total of *60,197* unique ICU visits, which are what this study refers to as 'patients'. The MIMIC-III dataset does not keep data on repeat patients, so the best effort that can be accomplished using this set is to treat every unique ICU visit as a unique patient.

A.1.2 Identify Sepsis Patients

Now that the data has been paired down to information related to the known patients' individual bodily states over time, it is necessary to determine which patients meet the Sepsis-3 definition[4] of Sepsis for use in the eventual model. In order to identify which patients will be used in the model, the Komoroski et. al research group provides a three step process to identify patients who have Sepsis in the MIMIC-III dataset[4]:

1. Flag patients with presumed infection
2. Compute SOFA at each of those patient's time-steps in the data

[4] https://github.com/matthieukomorowski/AI_Clinician/blob/master/AIClinician_sepsis3_def_160219.m

3. Flag Sepsis in Patients

The Sepsis indentifier script goes through the full set of patients located in the ABX data (which contains all admission related data for all ICU stays), and determines pairwise distances for bacteria culture presence in each patient at each time-step. If there is an increase in bacteria colony presence in 24 hours, the patient's onset of infection is marked at the timestamp of the beginning of the 24 hours, and they are flagged as a patient with infection. Of the initial *60,197* patients, this reduces the overall set down to *26,423* patients.

Recall from the background, the Sepsis-3 definition indicates that a patient has had an increase in their calculated SOFA score of 2 or more points. The next phase of the data filtering script takes the *26,423* patients and goes over all data points for each stay in the ICU. At each time-step, the patient's SOFA score is calculated based on the criteria for SOFA scores introduced in the background, and an additional column is appended with the calculated SOFA score for each patient. For every given patient's data, the script determines if there has been an increase in $\geq$2 SOFA points and flags that patient as a Sepsis patient. All of these IDs are then loaded into a separate data set, leaving the final dataset before full cleaning down from *26,423* patients to *21,463* patients.

A.1.3 Clean Data and Fill In Values

Komoroski et. al used several cleaning and data modification techniques throughout the entirety of their data processing pipeline, but the final stages of the pipeline are what clean and process the penultimate data set into the final set that will be used in the model.

All time stamp data for all ICU visits are modified to conform down into 4 hour blocks of time. Next, all ICU visits are shrunk to a 72 hour time window starting 24 hours before the onset of infection, and 48 hours after infection. There are a variable number of 4 hour *"blocs"* for each patient, depending on the length of their stay in the ICU. After collecting timestamps into blocs, the data filtering script eliminates outliers above and below certain threshold values, depending on their appearance in the dataset. A summary of this data can be found in Table A.2. Additionally, thresholds that are not present (upper or lower), are marked by a "-", indicating the data saw no outliers in that particular direction, or that outliers of those nature had already been removed by a

previous step.

After all outliers are removed, there are several computed fields that script generates. Specifically, any data column that revolves around a running total (*I.E.* Total Urine Output Produced) is calculated during this computed values phase. Following these computations, the script gathers all column headers and calculates which percentage of the data in each column is not missing (or NaN). If over 30% of the data for a given column is present, that column is kept, otherwise that column is removed from the dataset. After going through this factor removal, the dataset is left with the *59 columns* that will persist through the final stage of the pipeline. The final set of columns is available in Table A.1.

The final stage of the pipeline operates on all remaining rows and columns, of which accounts for a total of *21,463* patients. To account for the missing values in each column of data, the processing script makes use of the *knnimpute* function available through the MATLAB toolbox[5], in order to fill the missing values for each column of data based on the non-missing values. The *knnimpute* function in this script deploys the K-Nearest Neighbors algorithm using standardized (normalized) euclidean distances. All missing values per each column are imputed with a generated value using this method. After this process completes, all computed columns are recalculated with no missing data, yielding a final dataset that is at this point still in MATLAB's environment memory.

A.1.4 Rebuild MIMIC-III Using Only Sepsis Patients

One of the early goals of this research was to write the code in Python so that future machine learning researchers could make use of the work put in by this project and modify it as they saw fit. Since Python remains as the most popular language in the Computer Science for Machine Learning [6], Python was a logical option to take, in addition to its numerous convenience libraries for data manipulation and machine learning (NumPy, Pandas, etc..). Since Python is not built to interface directly with the MATLAB virtual environment, an export of some form needed to occur.

Fortunately, MATLAB provides a very easy way to convert a large table into a CSV file very

[5] https://www.mathworks.com/help/bioinfo/ref/knnimpute.html
[6] https://tinyurl.com/vnjszyb4.

quickly. Using the *writetable* method[7], MATLAB converts a MATLAB Table environment variable and writes out all data to a CSV file in a neatly formatted manner. The CSV file used as the final dataset, *patient_data.csv*, contains *238,331* rows, *21,463* patients, and *59* columns in total. The dataset can be rebuilt from the scripts discussed and linked previously, but the actual final data file cannot be provided anywhere due to the restrictions imposed on the MIMIC-III set from which it is built.

Bloc	ICU ID	Age
End of Record Delay	Weight	Systolic Blood Pressure
FiO2	Potassium	Glucose
CO2	SGOT	Albumin
INR	Arterial pH	Arterial BE
Max Dose Vasopressor	Fluid Input Total	Fluid Output 4 Hour
Chart Time	Gender	Elixhauser
GCS	HR	Mean Blood Pressure
Sodium	Chloride	BUN
SGPT	Bili	Hb
paO2	paCO2	Arterial Lactate
Fluid Input 4 Hours	Fluid Output Total	Cumulated Balance
Re-Admission	Died Within 48 Hours of Exit	Died In Hospital
Diastolic Blood Pressure	SpO2	Respiration Rate
Creatine	Calcium	Magnesium
WBC Count	PTT	Platelets
HCO3	Shock Index	Mechanical Ventilator
SOFA	PaO2_FiO2	SIRS
Died Within 90 days	Temperature	Ionized Calcium
PT	Median Dose Vasopressor	

Table A.1: All Factors in the Final Dataset [26]

[7] https://www.mathworks.com/help/matlab/ref/writetable.html

Column Data	Lower Threshold	Upper Threshold
Weight	-	300
Heart Rate	-	250
Systolic Blood Pressure	-	300
Avg. Overall Blood Pressure	0	200
Diastolic Blood Pressure	0	200
Respiration Rate	-	80
Oxygen Saturation (SpO2)	50	100
Body Temperature	25	90
Fraction of Inspired Oxygen (FiO2)	20	100
O2 Flow	-	75
Positive End Expiratory Pressure (PEEP)	0	40
Tidal Volume (TV)	-	1800
Minute Ventilation (MV)	-	50
Potassium (K+)	1	15
Sodium (Na)	95	178
Chloride (Cl)	70	150
Glucose (Glc)	1	1000
Creatine (Creat)	-	150
Magnesium (Mg)	-	10
Calcium (Ca)	-	20
Ionized Calcium	-	5
Carbon Dioxide (CO2)	-	120
Serum Glutamic Pyruvic Transaminase (SGPT)	-	10000
Serum Glutamic-Oxaloacetic Transminase (SGOT)	-	10000
Hemoglobin (Hb)	-	20
Height (Ht)	-	65
White Blood Cell Count (WBC)	-	500
Platelets (PLT)	-	2000
International Normalized Ratio (INR)	-	20
pH	6.7	8
Partial Pressure of Oxygen (PO2)	-	60
Partial Pressure of Carbon Dioxide (PCO2)	-	200
Bland Altman (BE)	-50	-
Lactate	-	30

Table A.2: Summary of all Threshold Values Used in Removing Outliers from Data [26]

Column	Type of Data	Column	Type of Data
Age	Normal	Arterial BE	Normal
Weight	Normal	HCO3	Normal
Glascow Coma Scale	Normal	Arterial Lactate	Normal
Heart Rate	Normal	SOFA	Normal
Systolic Blood Pressure	Normal	SIRS	Normal
Mean Blood Pressure	Normal	Shock Index	Normal
Diastolic Blood Pressure	Normal	PaO2 FiO2	Normal
Respiration Rate	Normal	Cumulated Balance	Normal
Temperature	Normal	qSOFA	Normal
FiO2	Normal	Gender	Binary
Potassium	Normal	Mechanical Ventilation	Binary
Sodium	Normal	Max Dose Vasopressor	Binary
Chloride	Normal	Re-Admission	Binary
Glucose	Normal	qSOFAFlag	Binary
Magnesium	Normal	SOFAFlag	Binary
Calcium	Normal	SpO2	Logarithmic
Hb	Normal	BUN	Logarithmic
WBC Count	Normal	Creatinine	Logarithmic
Platelets Count	Normal	SGOT	Logarithmic
PTT	Normal	SGPT	Logarithmic
PT	Normal	Bili	Logarithmic
Arterial pH	Normal	INR	Logarithmic
paO2	Normal	Fluid Input Total	Logarithmic
paCO2	Normal	Fluid Input 4 Hours	Logarithmic
Fluid Output Total	Logarithmic	Fluid Output 4 Hours	Logarithmic

Table A.3: All Factors Used In the Modeling Process

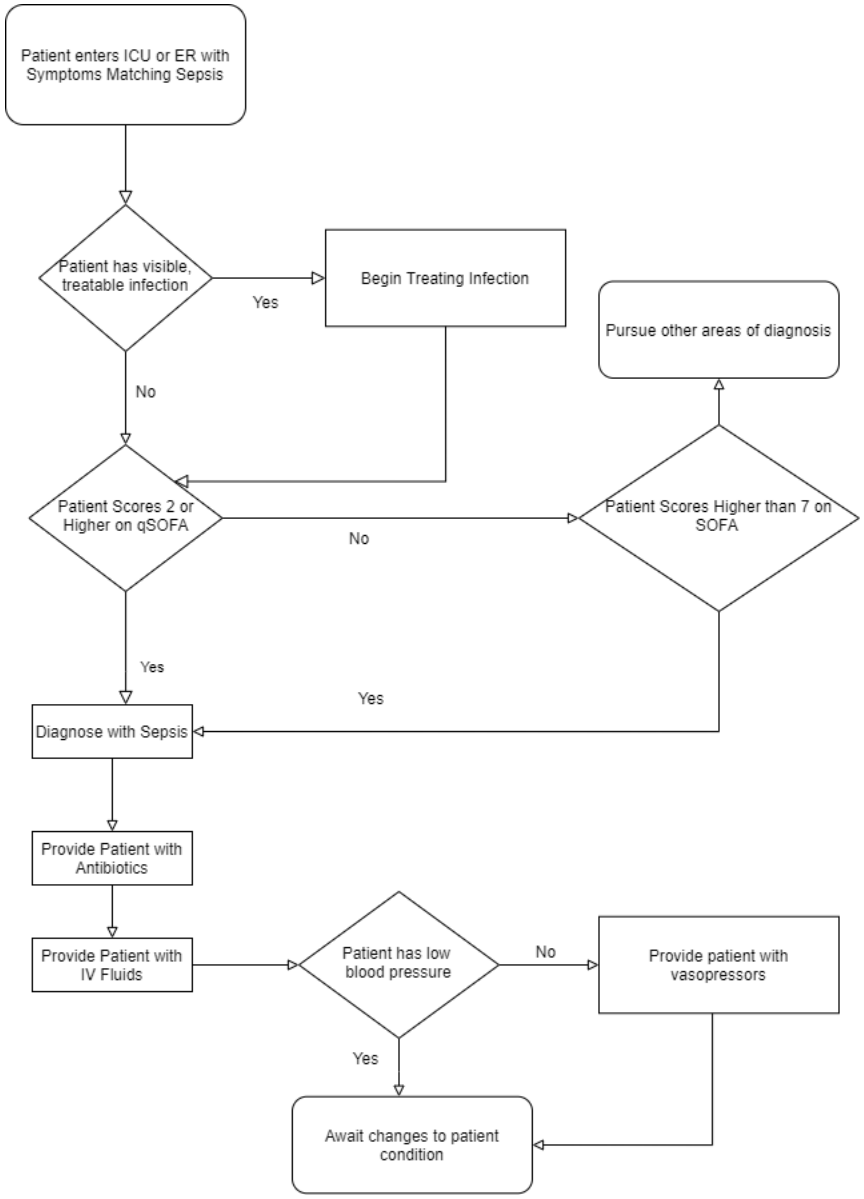


Figure A.1: A Flowchart for Understanding Sepsis Treatment and Diagnosis

References

[1] Brad Brown, Michael Chui, and James Manyika. Are You Ready for the Era of Big Data? *McKinsey Quarterly*, 2020.

[2] Andrew M. Nunn, Justin S. Hatchimonji, Daniel N. Holena, Mark J. Seamon, Brian P. Smith, Lewis J. Kaplan, Niels D. Martin, Patrick M. Reilly, C. William Schwab, and Jose L. Pascual. Boarding ICU patients: Are Our Rounding Practices Subpar? *The American Journal of Surgery*, 215(4):669–674, April 2018.

[3] K. Doi. Computer-Aided Diagnosis in Medical Imaging: Historical Review, Current Status and Future Potential. *Comput. Med. Imaging Graph*, 31(4-5):198–211, 2007.

[4] M. Singer, C. S. Deutschman, C. W. Seymour, M. Shankar-Hari, D. Annane, M. Bauer, R. Bellomo, G. R. Bernard, J. D. Chiche, C. M. Coopersmith, R. S. Hotchkiss, M. M. Levy, J. C. Marshall, G. S. Martin, S. M. Opal, G. D. Rubenfeld, T. van der Poll, J. L. Vincent, and D. C. Angus. The Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3). *JAMA*, 315(8):801–810, Feb 2016.

[5] CDC. What is Sepsis? *Centers for Disease Control and Prevention*, Aug 2019.

[6] Kristina E. Rudd, Sarah Charlotte Johnson, Kareha M. Agesa, Katya Anne Shackelford, Derrick Tsoi, Daniel Rhodes Kievlan, Danny V. Colombara, Kevin S. Ikuta, Niranjan Kissoon, Simon Finfer, Carolin Fleischmann-Struzek, Flavia R. Machado, Konrad K. Reinhart, Kathryn Rowan, Christopher W. Seymour, R. Scott Watson, T. Eoin West, Fatima Marinho, Simon I. Hay, Rafael Lozano, Alan D. Lopez, Derek C. Angus, Christopher J. L. Murray, and Mohsen

Naghavi. Global, Regional, and National Sepsis Incidence and Mortality, 19902017: Analysis for the Global Burden of Disease Study. *The Lancet*, 395(10219):200–211, January 2020.

[7] Greg S Martin, David M Mannino, Stephanie Eaton, and Marc Moss. The Epidemiology of Sepsis in the United States from 1979 through 2000. *New England Journal of Medicine*, 348(16):1546–1554, 2003.

[8] MedlinePlus. Septic Shock. *MedlinePlus Medical Encyclopedia*, Mar 2020.

[9] F. Balloux and L. van Dorp. Q&A: What are Pathogens, and What Have They Done to and for Us? *BMC Biol.*, 15(1):91, 10 2017.

[10] Djillali Annane, Eric Bellissant, and Jean-Marc Cavaillon. Septic Shock. *The Lancet*, 365(9453):6378, Jan 2005.

[11] M. H. Schoenberg, M. Weiss, and P. Radermacher. Outcome of Patients with Sepsis and Septic Shock after ICU Treatment. *Langenbecks Archives of Surgery*, 383(1):4448, Mar 1998.

[12] Martin Fraser Clark and Simon Victor Baudouin. A Systematic Review of the Quality of Genetic Association Studies in Human Sepsis. *Intensive Care Medicine*, 32(11):1706–1712, 2006.

[13] WebMD and Khatri Minesh. What is Sepsis or Septicemia (Blood Infection)? *WebMD*, May 2019.

[14] Krista O'Connell. Sepsis: Symptoms, Causes, Treatment, Risks & More. *Healthline*, Aug 2018.

[15] John C. Marshall. *Scoring Systems for Sepsis and the Multiple Organ Dysfunction Syndrome*, pages 921–931. Academic Press, Jan 2001.

[16] J. L. Vincent, R. Moreno, J. Takala, S. Willatts, A. De Mendonça, H. Bruining, C. K. Reinhart, P. M. Suter, and L. G. Thijs. The SOFA (Sepsis-related Organ Failure Assessment) Score to Describe Organ Dysfunction/Failure. *Intensive Care Medicine*, 22(7):707710, Jul 1996.

[17] A. E. Jones, S. Trzeciak, and J. A. Kline. The Sequential Organ Failure Assessment Score for Predicting Outcome in Patients with Severe Sepsis and Evidence of Hypoperfusion at the time of Emergency Department Presentation. *Crit Care Med*, 37(5):1649–1654, May 2009.

[18] CRISMA Center, University of Pittsburgh, and UPMC. qSOFA: quick Sepsis Related Organ Failure Assessment.

[19] J. R. Banegas, J. J. de la Cruz, F. Rodriguez-Artalejo, A. Graciani, P. Guallar-Castillin, and R. Herruzo. Systolic vs Diastolic Blood Pressure: Community Burden and Impact on Blood Pressure Staging. *J Hum Hypertens*, 16(3):163–167, Mar 2002.

[20] Royal College of Physicians and Surgeons of Glasgow. Assessment of Glasgow Coma Scale. *The Glasgow Structured Approach to Assessment of the Glasgow Coma Scale*.

[21] Sharath Nair, Anilkumar Surendran, Rajmohan Prabhakar, and Meer Chisthi. Comparison between FOUR score and GCS in Assessing Patients with Traumatic Head Injury: A Tertiary Centre Study. *International Surgery Journal*, 4:656, 01 2017.

[22] R. L. Gauer. Early Recognition and Management of Sepsis in Adults: the First Six Hours. *Am. Fam. Physician*, 88(1):44–53, Jul 2013.

[23] A. Kumar, D. Roberts, K. E. Wood, B. Light, J. E. Parrillo, S. Sharma, R. Suppes, D. Feinstein, S. Zanotti, L. Taiberg, D. Gurka, A. Kumar, and M. Cheang. Duration of Hypotension before Initiation of Effective Antimicrobial Therapy is the Critical Determinant of Survival in Human Septic Shock. *Crit. Care Med.*, 34(6):1589–1596, Jun 2006.

[24] CDC. Do You Know How Sepsis Is Diagnosed and Treated? *Centers for Disease Control and Prevention*, Jul 2019.

[25] Susan Duffy. Leading the Way in Treatment for Sepsis. *Emergency Services at Hasbro Childrens Hospital*, 2020.

[26] Matthieu Komorowski, Leo A. Celi, Omar Badawi, Anthony C. Gordon, and A. Aldo Faisal. The Artificial Intelligence Clinician Learns Optimal Treatment Strategies for Sepsis in Intensive Care. *Nature Medicine*, 24(11):1716–1720, October 2018.

[27] Matthew Reyna and Gari Clifford. Early Prediction of Sepsis from Clinical Data – the PhysioNet Computing in Cardiology Challenge 2019, 2019.

[28] Stephen Marsland. *Machine Learning: an Algorithmic Perspective*. Chapman and Hall/CRC Machine Learning and Pattern Recognition Series. CRC Press, second edition edition, 2015.

[29] Doina Precup, Richard S. Sutton, and Satinder Singh. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 759–766. Morgan Kaufmann, 2000.

[30] A. E. Johnson, M. M. Ghassemi, S. Nemati, K. E. Niehaus, D. A. Clifton, and G. D. Clifford. Machine Learning and Decision Support in Critical Care. *Proc IEEE Inst Electr Electron Eng*, 104(2):444–466, Feb 2016.

[31] J. H. Chen and S. M. Asch. Machine Learning and Prediction in Medicine - Beyond the Peak of Inflated Expectations. *N. Engl. J. Med.*, 376(26):2507–2509, Jun 2017.

[32] Zhongheng Zhang and written on behalf of AME Big-Data Clinical Trial Collaborative Group. Reinforcement Learning in Clinical Medicine: a Method to Optimize Dynamic Treatment Regime Over Time. *Annals of Translational Medicine*, 7(14), 2019.

[33] Yuan Ling, Sadid A. Hasan, Vivek Datla, Ashequl Qadir, Kathy Lee, Joey Liu, and Oladimeji Farri. Learning to Diagnose: Assimilating Clinical Narratives using Deep Reinforcement Learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 895–905, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

[34] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989.

[35] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, May 1992.

[36] Bob Givan and Ron Parr. An Introduction to Markov Decision Processes. *Exploration of Large State Spaces*, Nov 2001.

[37] David Silver. Markov Decision Processes. *UCL - Londons Global University*, Dec 2018.

[38] P. Mehta and S. Meyn. Q-learning and Pontryagin's Minimum Principle. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 3598–3605, 2009.

[39] Venelin Valkov. Solving an MDP with Q-Learning from scratch Deep Reinforcement Learning for Hackers (Part 1). *Medium*, Apr 2019.

[40] Avinash K. Dixit. *Optimization in Economic Theory*. Oxford Univ. Press, 2. ed, paperback ed. repr. 1997 edition, 1997.

[41] Chao Yu, Jiming Liu, and Shamim Nemati. Reinforcement Learning in Healthcare: A Survey. 2019.

[42] Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. Guidelines for Reinforcement Learning in Healthcare. *Nature Medicine*, 25(1):16–18, 2019.

[43] Bradley Efron. Trevor Hastie. *Computer Age Statistical Inference*. Cambridge University Press, 2016.

[44] Pat Bartlein. Geographic Data Analysis. *University of Oregon*, 2018.

[45] R. E. Kass. Statistical Inference: The Big Picture. *Stat Sci*, 26(1):1–9, Feb 2011.

[46] Subrata Kumar Das. *Computational Business Analytics*. Chapman and Hall/CRC, 2014.

[47] Judea Pearl and Dana Mackenzie. *The Book of Why: The Book of Cause and Effect*. Basic Books, first edition, 2018.

[48] S. A. Baldwin, E. Stice, and P. Rohde. Statistical Analysis of Group-Administered Intervention Data: Reanalysis of Two Randomized Trials. *Psychother Res*, 18(4):365–376, Jul 2008.

[49] P. E. Lekone and B. F. Finkenst. Statistical Inference in a Stochastic Epidemic SEIR Model with Control Intervention: Ebola as a Case Study. *Biometrics*, 62(4):1170–1177, Dec 2006.

[50] Ken'ichi Morooka. A Survey on Statistical Modeling and Machine Learning Approaches to Computer Assisted Medical Intervention: Intraoperative Anatomy Modeling and Optimization of Interventional Procedures. *IEICE Transactions on Information and Systems*, E96-D:pp.784–797, 04 2013.

[51] Matthew A. Reyna, Christopher S. Josef, Russell Jeter, Supreeth P. Shashikumar, M. Brandon Westover, Shamim Nemati, Gari D. Clifford, and Ashish Sharma. Early Prediction of Sepsis From Clinical Data. *Critical Care Medicine*, 48(2):210–217, February 2020.

[52] J. Morrill, A. Kormilitzin, A. Nevado-Holgado, S. Swaminathan, S. Howison, and T. Lyons. The Signature-Based Model for Early Detection of Sepsis From Electronic Health Records in the Intensive Care Unit. In *2019 Computing in Cardiology (CinC)*, pages 1–4, 2019.

[53] Aniruddh Raghu, Matthieu Komorowski, Imran Ahmed, Leo A. Celi, Peter Szolovits, and Marzyeh Ghassemi. Deep Reinforcement Learning for Sepsis Treatment. *CoRR*, abs/1711.09602, 2017.

[54] Aniruddh Raghu, Matthieu Komorowski, Leo Anthony Celi, Peter Szolovits, and Marzyeh Ghassemi. Continuous State-Space Models for Optimal Sepsis Treatment - a Deep Reinforcement Learning Approach. *CoRR*, abs/1705.08422, 2017.

[55] Nan Jiang and Lihong Li. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. *ICML*, 2016.

[56] X. Peng, Y. Ding, D. Wihl, O. Gottesman, M. Komorowski, L. H. Lehman, A. Ross, A. Faisal, and F. Doshi-Velez. Improving Sepsis Treatment Strategies by Combining Deep and Kernel-Based Reinforcement Learning. *AMIA Annu Symp Proc*, 2018:887–896, 2018.

[57] Brenden K. Petersen, Jiachen Yang, Will S. Grathwohl, Chase Cockrell, Claudio Santiago, Gary An, and Daniel M. Faissol. Precision Medicine as a Control Problem: Using Simulation and Deep Reinforcement Learning to Discover Adaptive, Personalized Multi-Cytokine Therapy for Sepsis. *CoRR*, abs/1802.10440, 2018.

[58] Athanasios Tsoukalas, Timothy Albertson, and Ilias Tagkopoulos. From Data to Optimal Decision Making: A Data-Driven, Probabilistic Machine Learning Approach to Decision Support for Patients With Sepsis. *JMIR Med Inform*, 3(1):e11, Feb 2015.

[59] N. Hou, M. Li, L. He, B. Xie, L. Wang, R. Zhang, Y. Yu, X. Sun, Z. Pan, and K. Wang. Predicting 30-days Mortality For MIMIC-III Patients with Sepsis-3: a Machine Learning Approach using XGboost. *J Transl Med*, 18(1):462, 12 2020.

[60] Guilan Kong, Ke Lin, and Yonghua Hu. Using Machine Learning Methods to Predict In-Hospital Mortality of Sepsis Patients in the ICU. *BMC Medical Informatics and Decision Making*, 20(1):251–251, Oct 2020. 33008381[pmid].

[61] Matthew A. Reyna, Christopher S. Josef, Russell Jeter, Supreeth P. Shashikumar, M. Brandon Westover, Shamim Nemati, Gari D. Clifford, and Ashish Sharma. Early prediction of sepsis from clinical data: The physionet/computing in cardiology challenge 2019. *Critical Care Medicine*, 48(2), 2020.

[62] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, A Freely Accessible Critical Care Database. *Scientific Data*, 3(1), May 2016.

[63] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer, First edition, 2017.

[64] S. Yadav and S. Shukla. Analysis of K-Fold Cross-Validation Over Hold-Out Validation on Colossal Datasets for Quality Classification. In *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, pages 78–83, 2016.

[65] A. Fisher, B. Adhikari, C. Zhai, J. E. Morgan, V. K. Mago, and P. J. Giabbanelli. Predicting The Resource Needs And Outcomes of Computationally Intensive Biological Simulations. pages 1–12, 2020.

[66] Yaqian Zhang, Jacek Mańdziuk, Chai Hiok Quek, and Boon Wooi Goh. Curvature-Based Method for Determining the Number of Clusters. *Information Sciences*, 415:414–428, 2017.

[67] Nicholas Rosso and Philippe Giabbanelli. Accurately inferring compliance to five major food guidelines through simplified surveys: Applying data mining to the uk national diet and nutrition survey. *JMIR Public Health Surveill*, 4(2):e56, May 2018.

[68] Venkata Sai Pillutla, Andrew A. Tawfik, and Philippe J. Giabbanelli. Detecting the Depth and Progression of Learning in Massive Open Online Courses by Mining Discussion Data. *Technology, Knowledge and Learning*, 25(4):881–898, February 2020.

[69] A. Awad, M. Bader-El-Den, J. McNicholas, J. Briggs, and Y. El-Sonbaty. Predicting hospital mortality for intensive care unit patients: Time-series analysis. *Health Informatics J*, 26(2):1043–1059, 06 2020.

[70] T. Sinuff, N. K. Adhikari, D. J. Cook, H. J. Schünemann, L. E. Griffith, G. Rocker, and S. D. Walter. Mortality predictions in the intensive care unit: comparing physicians with scoring systems. *Crit Care Med*, 34(3):878–885, Mar 2006.

[71] Angela K. M. Lipshutz, John R. Feiner, Barbara Grimes, and Michael A. Gropper. Predicting Mortality in the Intensive Care Unit: a Comparison of the University Health Consortium Expected Probability of Mortality and the Mortality Prediction Model III. *Journal of Intensive Care*, 4(1):35, May 2016.