The Medication Ordering Safety System:

A Pipeline to Predict Adverse Drug Events at the Time of Ordering

By

Aileen Wright, M.D.

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Biomedical Informatics

May 14, 2021

Nashville, Tennessee

Approved:

Dara Mize, M.D., M.S.

Joshua Smith, Ph.D.

Scott Nelson, Pharm.D., M.S.

DEDICATION

To Adam, Isaac, Grace, and someone else who will have a name very soon!

ACKNOWLEDGEMENTS

First, I would like to thank my thesis mentor, Dara Mize, who was an incredible advisor and support throughout this process. Thank you for your thoughtful guidance from day one, and for helping this project become what it is today. Thank you to my thesis committee members Josh Smith and Scott Nelson who provided valuable and transformative feedback.

I am grateful for the opportunity to have studied in the Department of Biomedical Informatics here at Vanderbilt. I appreciate the leadership of the master's program, including Kim Unertl and Cindy Gadd. Thank you to Rischelle Jenkins for advice, support, and behind-the-scenes work throughout this process. Thank you to our outstanding chair, Kevin Johnson, for inspiring all of us to pursue our dreams, research-related and otherwise.

I am indebted to the ACCRE team, without whom I could not have completed this work. I am also grateful to the Clarity team, who maintains a resource so important for operations and innovation at our hospital.

I am so grateful to my patients, who remind me why we must work to innovate and improve the systems of healthcare.

This work would never have existed without the incredible support of my family. I had a fantastic person to bounce ideas off of in Adam Wright, my husband. I am so grateful for our children, Isaac and Grace, for bringing joy into our lives.

Supported by the National Library of Medicine of the National Institutes of Health under Award Number T15LM007450.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

Introduction

The past few decades have seen a dramatic transformation of the healthcare system as hospitals have overwhelmingly adopted vendor-based electronic health records (EHRs), completely changing the practice of medicine for physicians[1]. Healthcare systems have struggled with rising costs and transitioned to value-based care[2]. Medical knowledge has continued to advance at a rapid pace, leading to the generation of clinical guidelines which may already be outdated by the time clinical decision support (CDS) systems are built. Clinicians struggle to keep up with medical knowledge. The promises of EHRs, such as improved quality and safety, have not yet been fully realized, even as large amounts of "big data" are generated through clinical care using EHR systems[3]. Meanwhile, technology has advanced considerably outside of the medical field, especially in the area of machine learning, where techniques like deep learning are shaping the future of artificial intelligence[4].

In the everyday practice of medicine on the hospital floor, it can seem as though the advances of modern technology have not yet permeated through the hospital walls. Medical errors still abound[5], and the computer can often seem more like a foe to the clinician than a helpful assistant[6]. One of the most frequent nuisances clinicians experience throughout the day is that of CDS alerts which are unhelpful and must be overridden for the clinician to proceed with his or her work[7].

Even as clinical data amasses and machine learning provides an opportunity to build more useful CDS, the rate of progress is slow, and most alerts continue to be simple, such as those that warn about potential interactions between medications[8]. Few CDS alerts make use of machine learning to incorporate multiple patient characteristics and provide better guidance to clinicians. As we will describe, machine learning-based alerts have the potential to perform better than simple alerts, provide more useful information to the clinician, and decrease the rate of "nuisance alerts," but they are challenging to develop. To combat the slow pace of innovation in healthcare[9], we need to develop more effective systems and technologies for learning from data, making use of clinicians and informaticians throughout this process.

Here, we present the Medication Ordering Safety System (MOSS), a platform we designed to improve the process of learning from data to generate machine learning-based CDS around medication safety. By streamlining the process of predictive modeling, we aim to create CDS which performs better than the typical alerts that fill today's EHR. By removing technical barriers, we aim to develop a system which is more approachable to clinicians and other personnel who could not otherwise contribute to the process of learning from data to generate more intelligent CDS. Rapid generation, updating and retraining of clinical prediction models through such a pipeline could help address the issue of calibration drift[10]. Such a system could also help institutions train clinical prediction models using their own data rather than relying on models developed at outside institutions which may not be relevant to their own population.

In the chapters that follow, we present the motivation for creating MOSS, describe its design, and provide examples of its use. Chapter 2 describes the potential benefits and challenges of using machine learning to extract knowledge from clinical data to inform improved CDS. Chapter 3 outlines the architecture of MOSS, which standardizes the process of generating predictive models around medication safety to make it more approachable for novices, as well as more efficient for advanced data scientists. In chapters 4 and 5, we present two clinical scenarios for which we used MOSS to train prediction models: one to predict low blood pressure in hospitalized patients after blood pressure-lowering medications are prescribed, and another to predict low blood glucose after insulin is prescribed. Finally, in chapter 6 we discuss the strengths and limitations of MOSS and describe our future research directions as we build out the MOSS pipeline and improve its ability to catalyze the creation of machine learning-based CDS.

CHAPTER 2

Background

2.1. Introduction

The idea of learning from clinical data to improve patient safety is not a new one. In this chapter, we introduce the problem of medication errors, the potential benefits of EHRs and computerized prescribed order entry (CPOE) in improving patient safety, and reasons for the discrepancy in the perceived benefits of these technologies and the persistence of medication errors in modern healthcare. Next, we will discuss the potential benefits and challenges of using machine learning to improve CDS around medication safety. Finally, we discuss current technologies that help researchers learn from clinical data and provide motivation for developing tools like MOSS which might facilitate the process of learning from clinical data to improve CDS.

2.2. Medication Errors

In the 1990s, researchers reported that hundreds of thousands of people die in the US each year due to medical injuries and that most medical injuries are "adverse drug events" caused by medication errors[11]. These adverse drug events were found to cost hospitals millions of dollars per year. Notably, a large proportion of these were preventable errors[12]. In 1999, in "To Err is Human: Building a Safer Health System," the Institute of Medicine reported that an estimated 44,000-98,000 people die each year due to preventable medical errors[13]. The same institute published recommendations in 2007 on how to prevent medication errors, including strategies such as creating a culture of safety, ensuring safe nursing ratios, involving pharmacists, improving patient identification, and, importantly, implementing CPOE[14].

2.3. Benefits of CPOE in Preventing Medication Errors

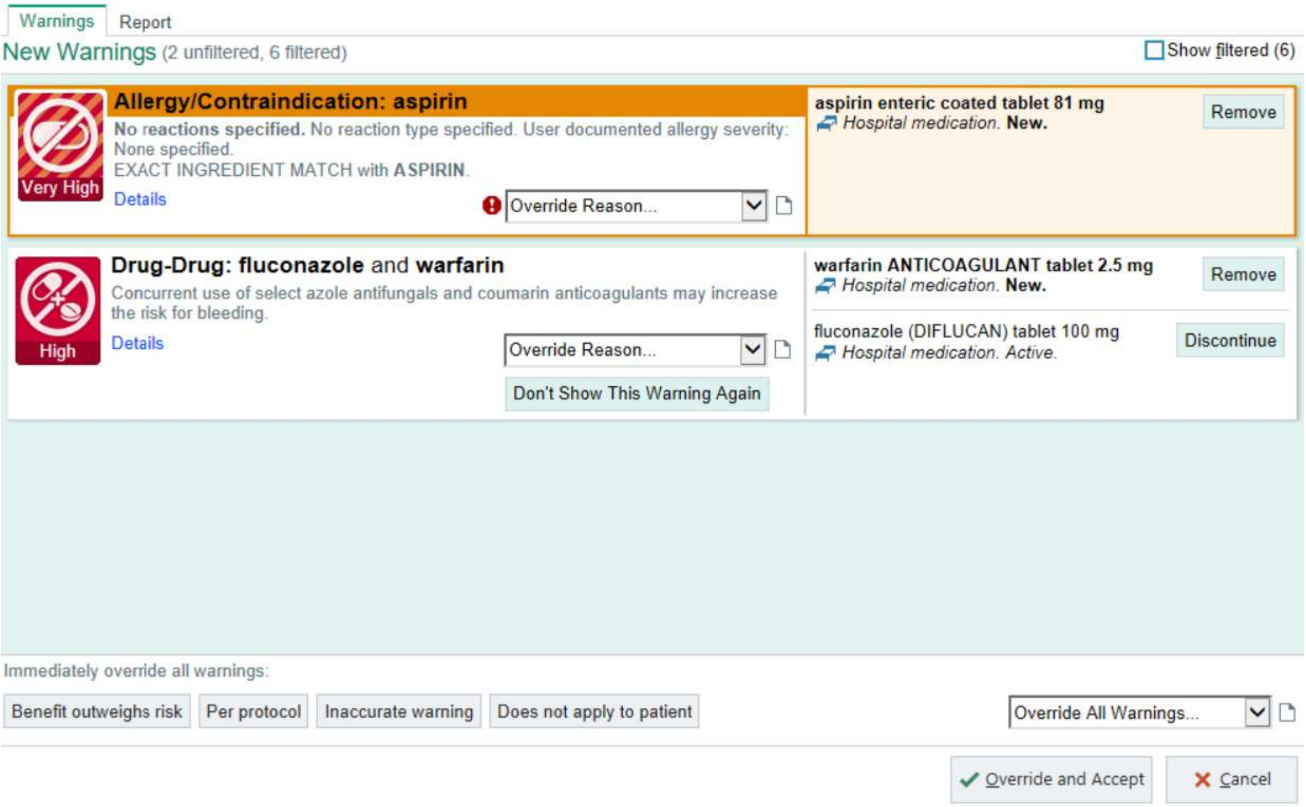
The promotion of CPOE as a solution for medication errors was based on work by researchers like David Bates who had found that computer order entry decreased serious medication errors by 55%[15]. Rather than relying on hand-written orders, CPOE requires prescribers to use a computer interface to enter orders[16]. One study found that implementing CPOE and an electronic medication administration record eliminated all physician and nursing transcription errors[17]. In addition to reducing risks of illegible prescriptions, saving time for clinicians, and saving costs by reducing the need for paper forms, CPOE provides the opportunity for CDS to be integrated into the medication ordering process. For example, CPOE with effectively designed CDS could help ensure that the medication matches the patient's condition, is the correct dose, is appropriately adjusted for a patient's age and kidney function, and is not on the patient's allergy list[18].

2.4. The Gap in Perceived Benefits and Actual Outcomes

Since then, major healthcare systems have switched from "home-grown" EHRs to commercial vendor systems which come with built-in medication alerts and other CDS. After the Health Information Technology for Economic and Clinical Health (HITECH) Act of 2009, EHRs became widely adopted[1]. With increased use of CPOE, one might expect the problem of medication errors to be largely solved. On the contrary, in 2016, Makary et al. suggested that medical errors may now be the third leading cause of death in the United States[5].

2.5. Table-based medication alerts

The gap in the perceived benefit of CPOE and the progress to date in reducing medical errors likely relates to the state of clinical decision support used in modern EHRs. Medication alerts are most commonly rule-based alerts which rely on tables, such as those pertaining to allergies, potential interactions between medications, or duplicative therapy[8].



New Warnings (2 unfiltered, 6 filtered)

Allergy/Contraindication: aspirin — Very High

No reactions specified. No reaction type specified. User documented allergy severity: None specified.
EXACT INGREDIENT MATCH with **ASPIRIN**.

aspirin enteric coated tablet 81 mg
Hospital medication. **New.**

Drug-Drug: fluconazole and **warfarin** — High

Concurrent use of select azole antifungals and coumarin anticoagulants may increase the risk for bleeding.

warfarin ANTICOAGULANT tablet 2.5 mg
Hospital medication. **New.**

fluconazole (DIFLUCAN) tablet 100 mg
Hospital medication. Active.

Immediately override all warnings:

Figure 1. Example of table-based medication alerts. © 2021 Epic Systems Corporation.

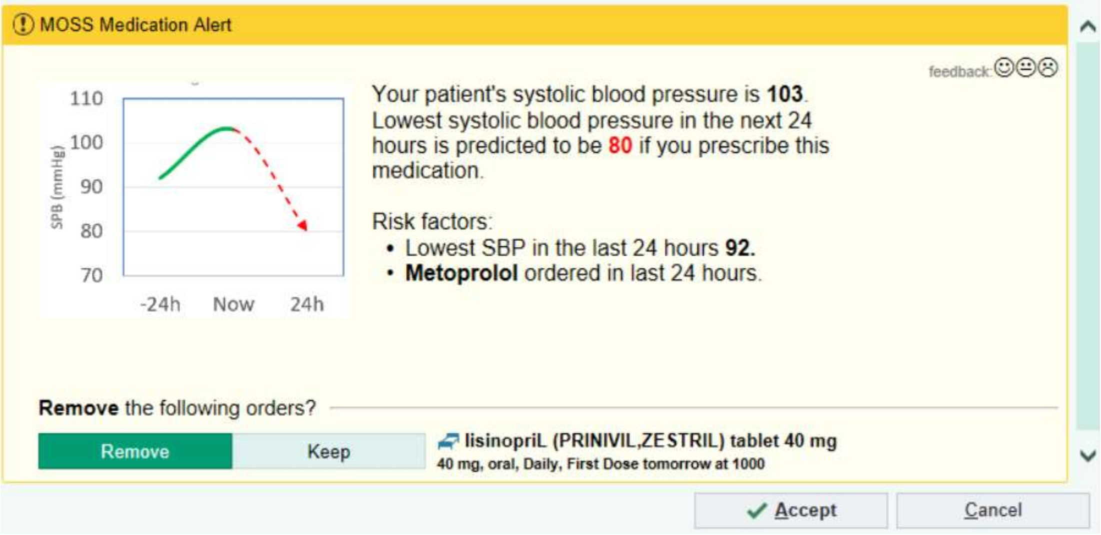
For example, let us imagine a scenario where a clinician is ordering aspirin and warfarin for her patient. She may be displayed two medication alerts (Figure 1). The first alert warns that she is prescribing aspirin to a patient with an aspirin allergy. The second alert is a drug-drug interaction alert which warns her about a potential interaction between warfarin, which she is trying to prescribe, and fluconazole, another medication the patient is already taking. While these are quite different alerts, they both have in common that they are triggered using tables and simple rules. The aspirin allergy alert uses a table of all the medications the patient is allergic to. If the new medication being prescribed has an ingredient matching one on the patient's allergy list, an alert is triggered. Similarly, the drug-drug interaction alert depends on a table which lists all medications that interact with warfarin and is triggered if the clinician tries to prescribe warfarin to a patient on one of these medications. Such table-based alerts do not account for patient-specific characteristics such as the patient's most recent laboratory results, vital signs, or actual bleeding risk. Occasionally, alerts may incorporate basic patient information, such as age or kidney function but these are less common than simple rule and table-based alerts. Research has shown that simple displays of laboratory data alongside alerts may not be helpful enough[19]. The National Academy of Medicine has highlighted the importance of leveraging multiple data types for successful clinical decision support[20].

2.6. Alert Fatigue

Studies of medication alerts have revealed some of the many challenges of building clinical decision support that is effective and helpful to clinicians[7]. Alert overrides frequently occur, such as when a patient receives an alert about a drug-drug interaction when prescribing warfarin but prescribes it anyway. Alerts may contribute to "alert fatigue," where clinicians who are repeatedly shown warnings become less likely to pay attention to them. Alert fatigue can contribute to users' dissatisfaction with the EHR, which may increase clinician burnout[21]. The low positive predictive value of table-based alerts likely contributes to alert fatigue and frequent alert overrides. For example, one common type of medication alert warns clinicians of the risk of torsades de pointes, a dangerous heart rhythm, when ordering medications which prolong the QT interval[22]. However, a real-world analysis looking at how often propofol administration really leads to torsades de pointes found a very low incidence of 1.93 per million[23]. In our analysis, this would correspond to a precision of 0.002% and sensitivity of 38% for drug-drug interaction alerts warning of the potential additive effect between propofol and other medications which also prolong the QT interval. Even very low-performing machine learning-based prediction models have much higher performance, demonstrating the potential of machine learning to improve upon the low performance of table-based alerts.

2.7. Clinical Prediction Models

Risk prediction models have been developed for a wide range of uses in healthcare. For example, there are a multitude of calculators to predict cardiovascular risk, of which the most commonly used is the 2013 American College of Cardiology/American Heart Association Pooled Cohort Equations CV Risk Calculator[24]. The pulmonary embolism rule-out criteria are used to identify patients who may not need further testing for pulmonary embolism[25]. The Morse Fall Scale is commonly used in the acute hospital setting to predict falls[26]. Another calculator, the FIB-4 index, is used to predict advanced fibrosis in patients with concern for liver disease[27]. While historically, the traditional approach to developing clinical risk prediction models has been to use regression models such as logistic regression, more recently, machine learning algorithms such as neural networks, support vector machines, and random forest have been used[28]. Innovations in machine learning and predictive modeling present us with an opportunity to develop more intelligent machine learning-based clinical decision support systems[29]. Such systems would make use of "big data" to generate predictive models which can be integrated into the EHR to prevent medical errors.



Your patient's systolic blood pressure is **103**. Lowest systolic blood pressure in the next 24 hours is predicted to be **80** if you prescribe this medication.

Risk factors:

- Lowest SBP in the last 24 hours **92.**
- **Metoprolol** ordered in last 24 hours.

Remove the following orders?

lisinopriL (PRINIVIL,ZESTRIL) tablet 40 mg
40 mg, oral, Daily, First Dose tomorrow at 1000

Figure 2. A mock-up of a machine learning-based medication alert. © 2021 Epic Systems Corporation.

2.8. Machine learning-based clinical decision support systems

As an example, let us consider a clinical scenario where a physician is prescribing a blood pressure-lowering medication to a patient with a normal blood pressure. A machine-learning based alert (Figure 2), trained using data from the hospital's clinical data warehouse, could incorporate multiple patient factors to predict blood pressure, such as the patient's diagnoses, vital signs, laboratory values, and medications.

Such an alert could show the user patient-specific factors that put the patient at risk for low blood pressure after the medication is prescribed. By making use of clinical data from the hospital's own population, such alerts have the potential to have improved positive predictive value and sensitivity than table-based alerts.

2.9. Integration of Clinical Prediction Models into the EHR

The use of machine learning and predictive models to generate CDS may seem intuitive, with clear benefits. However, while a significant number of machine learning algorithms and predictive models have been developed, evaluated, and published, few have been successfully integrated into the EHR as clinical decision support systems.

Some notable exceptions exist. One group implemented a real-time sepsis alert for patients admitted to a medicine ward, and found that 70.8% of patients in the intervention group received at least one intervention for sepsis, as compared to 55.8% of patients in the nonintervention group (p=0.18)[30]. Another group implemented a readmission prediction tool in their EHR and found an area under the curve of 0.716-0.760[31]. The New York Department of Health integrated a fall prevention program into the EHRs of primary care sites and saw as many as 79.0% of eligible patients screened for fall risk in the first 12 months of their study[32]. However, the few models that have been integrated into the EHR may perform poorly[33]. Sittig et al. named effective mining of large clinical databases to create new CDS one of the grand challenges of CDS and highlighted the idea that new clinical knowledge, guidelines, and CDS interventions are waiting to be developed if we can find ways to overcome this grand challenge[34].

2.10. Challenges of learning from clinical data to create CDS

The process of developing clinical prediction models for CDS is challenging. Each step in the process – determining the question the model proposes to address, preparing the theoretical framework, selecting the dataset, determining which variables to include, generating the prediction models, and evaluating their performance – has practical barriers[35]. First, the developer must have access to data – a not insignificant task due to protections around health data. This data must then be extracted from the EHR or back-end database to build and train the model. Clinicians often need to work with non-clinician data analysts to extract relevant data from the clinical data warehouse[36]. This process is often time-consuming, as clinical concepts must be abstracted to rules, and knowledge must be transferred back and forth from the clinician to the technical expert.

The actual task of data extraction typically requires knowledge of SQL, the data warehouse's structure, and the specific tables with the data of interest. Features must be engineered and specified in detailed queries.

For example, it is not enough to specify that one is looking for patients with diabetes. One must decide whether to use the patient's problem list, billing diagnosis codes, and/or laboratory values, such as hemoglobin A1c. Entire research labs are devoted to developing these kind of phenotyping algorithms.

Beyond diagnosis, other potential features such as medications, lab results, and vital signs can be similarly complex. Use of medications as features requires knowing at what level to categorize them, whether it be the pharmaceutical class, subclass, generic medication, or brand name. One must decide whether to include medications which combine more than one ingredient, such as oxycodone and acetaminophen, and how to classify them: should this count as an opioid or an anti-pyretic? Should only ordered medications be used as features, or also administered medications? Such decisions require both knowledge of clinical concepts and workflow, and the realities of the clinical data warehouse at a granular level. Moreover, after the features are engineered and the data extracted, it must be cleaned, pre-processed, formatted, and loaded into a platform where predictive modeling can be performed and evaluated.

Further challenges surround the process of integrating such models into the EHR for CDS. Additionally, models that are developed at one institution may not perform as well at other institutions[37] and making models generalizable may risk sacrificing performance at any one institution[38]. Those models which are implemented may suffer from calibration drift, in which model accuracy deteriorates over time due to dynamic clinical environments[39]. One study of models for hospital-acquired acute kidney injury over 9 years showed that calibration declined and models increasingly overpredicted the outcome[10]. The process of extracting clinical data, building predictive models, and integrating models into EHR as CDS takes months, if not longer, which may explain the relative paucity of intelligent clinical decision support systems in today's EHRs.

2.11. Query Tools

Given the time-intensive nature of the data extraction process, certain tools have been developed to help researchers query clinical databases directly in more user-friendly interfaces[34]. One example of this is the Partners Research Patient Data Registry (RPDR) which was developed at Partners Healthcare in Boston, MA[40]. The RPDR is a drag-and-drop web query tool which helps users determine, for example, how many patients within the clinical data warehouse have both a certain diagnosis and a certain lab value. Users may request data through this tool, and it has proved useful for developing clinical trial cohorts. Integrating Informatics and Biology at the Beside (i2b2) uses a similar strategy of a user-friendly interface for creating queries and has been used at a wide network of clinical research institutions[41(p2)]. Vanderbilt has a similar tool in Record Counter, which allows researchers to query a de-identified version of the clinical data warehouse using a visual interface and request datasets[42]. Additional query tools for clinical data have been developed by the Observational Health Data Sciences and Informatics program (OHDSI) which creates tools centered around the Observational Medical Outcomes Partnership Common Data Model (OMOP CDM)[43]. However, no self-service query tool that we know of has been developed with the purpose of facilitating predictive modeling around medication safety.

2.12. Conclusion

The adoption of EHRs and CPOE most likely has improved the safety and quality of today's practice of medicine. However, as medicine becomes more complex, new medications are developed, and the amount of clinical knowledge that each clinician is expected to master expands, the task of driving out medical error becomes more difficult. Machine learning presents us with an opportunity to address these challenges.

The work of improving CDS has the potential to deliver not only safer and higher-quality patient care, but also a patient care process which is more enjoyable for clinicians, who are currently bombarded with low positive predictive value alerts. Developing and adopting machine learning-based CDS will require improved integration of clinical experts into the predictive modeling process. Learning from clinical data to create advanced CDS may be stalled until we develop effective tools to make this process more efficient and approachable for large numbers of clinicians and informaticians.

CHAPTER 3

Design of the Medication Ordering Safety System

3.1. Introduction

This chapter outlines the architecture of MOSS and its various modules which support the user in accessing the clinical data warehouse and training and evaluating predictive models. We describe the individual modules of MOSS. We also use an example to show how MOSS can be used to generate a predictive model. Finally, we discuss MOSS' unique way of approaching diagnoses and developing medication scores for predictive modeling.

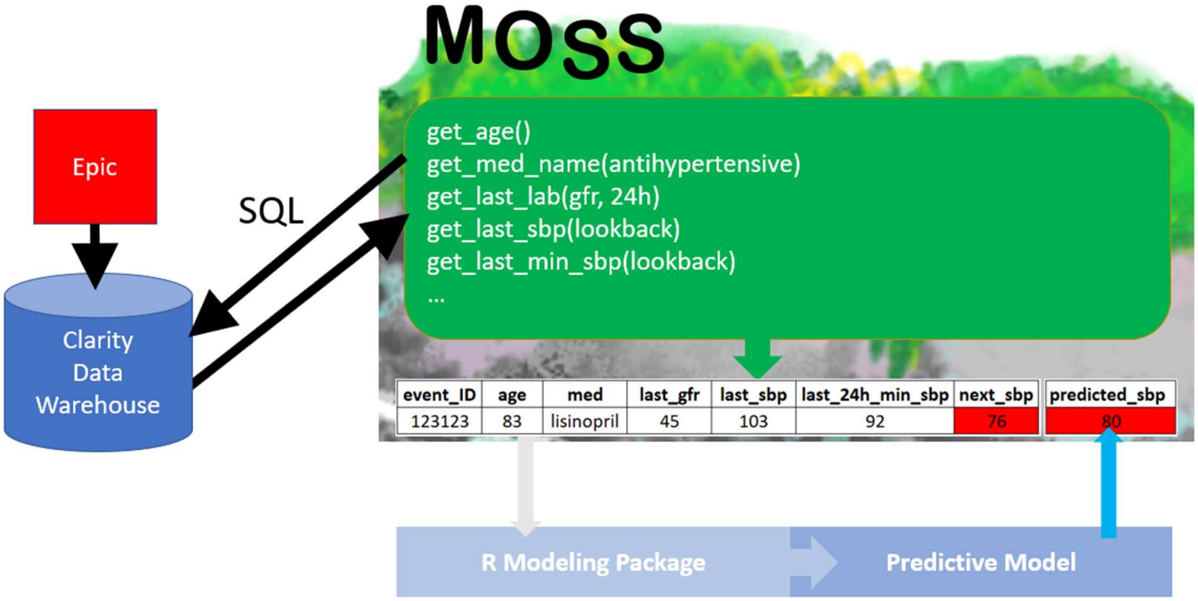
3.2. Motivation

The process of using machine learning to develop clinical prediction models can be resource-intensive and time-consuming. Extracting data from the clinical data warehouse typically requires knowledge of SQL. Seemingly simple tasks such as extracting lab values for patients admitted to the hospital during a certain time period can require multiple lines of code. Even for those already experienced with writing SQL queries, each new data extraction task may require remembering minute details about which table contains certain data. Query writers often save old queries and copy and paste them into a new query, before modifying them for the task at hand. Such a process takes time. We designed MOSS to run entirely within R and allow the user to extract data by making use of simple functions. SQL queries are generated automatically by MOSS functions, allowing the user to more easily and quickly extract data from the clinical data warehouse.

3.3. Pipeline design

The design of MOSS is shown in Figure 3. Data generated through the care of patients using VUMC's EHR, Epic, is saved in the Clarity Data Warehouse (© 2021 Epic Systems Corporation). MOSS runs entirely in the R programming language. Data extraction functions are used to generate SQL to query

Clarity and return features in the form of tabular data. These data are combined into a dataset which is used to train and test models.



event_ID	age	med	last_gfr	last_sbp	last_24h_min_sbp	next_sbp	predicted_sbp
123123	83	lisinopril	45	103	92	76	80

Figure 3. Medication Ordering Safety System diagram. Data generated through patient care is stored in the Clarity data warehouse. MOSS generates SQL queries which return results that are joined into a data structure used by modeling packages in R to generate predictions.

3.4. Modules

MOSS contains five modules, including a cohort creation module, feature extraction module, dataset creation module, model training module, and model evaluation module (Figure 4). This modular design allows for flexibility: if a user prefers not to utilize the cohort creation module, he or she may instead substitute his or her own cohort which can be used with the feature extraction module. Similarly, once a user has used the dataset creation module to generate a dataset, this dataset may be used with any standard modeling package in R rather than MOSS’ model training and evaluation modules.

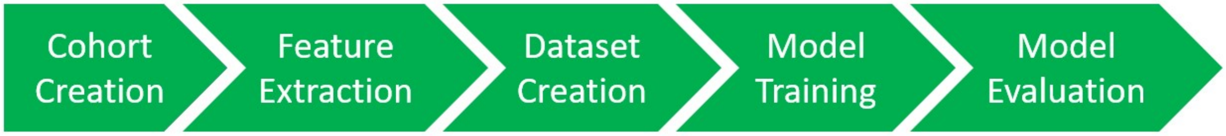
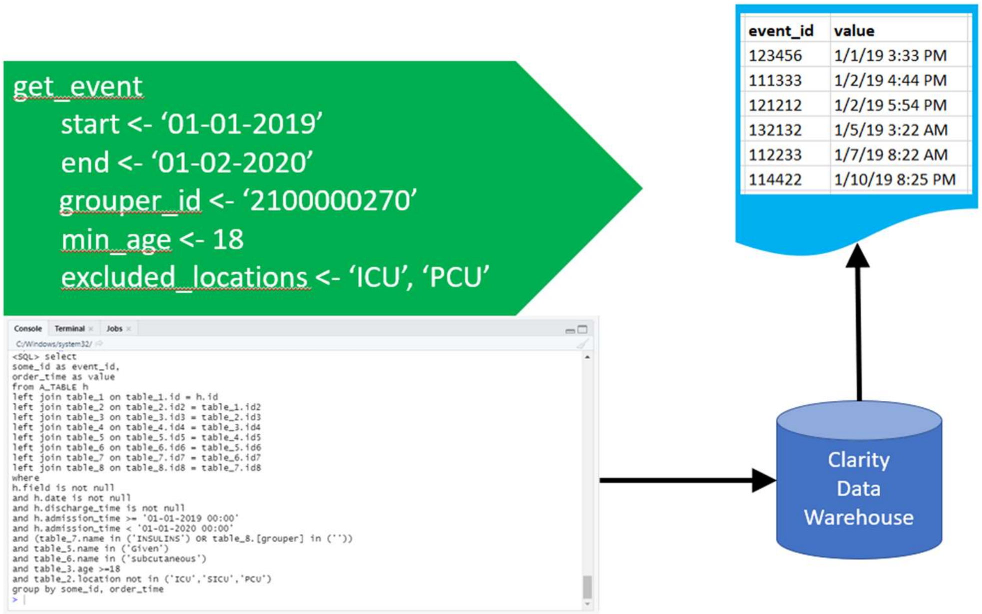


Figure 4. Medication Ordering Safety System modules.

3.5. MOSS Example

In the following sections, we will describe the five modules of MOSS. To illustrate how these modules work, we will use the example of Dr. Green, who uses MOSS to generate a predictive model about low blood sugar. She wants to predict whether an insulin order will be followed by a low blood glucose within the 24 hours following the order. She will use two features to train her model: the patient's weight, and the patient's last known blood glucose before the new insulin order.


```
get_event
     start <- '01-01-2019'
     end <- '01-02-2020'
     grouper_id <- '2100000270'
     min_age <- 18
     excluded_locations <- 'ICU', 'PCU'
```

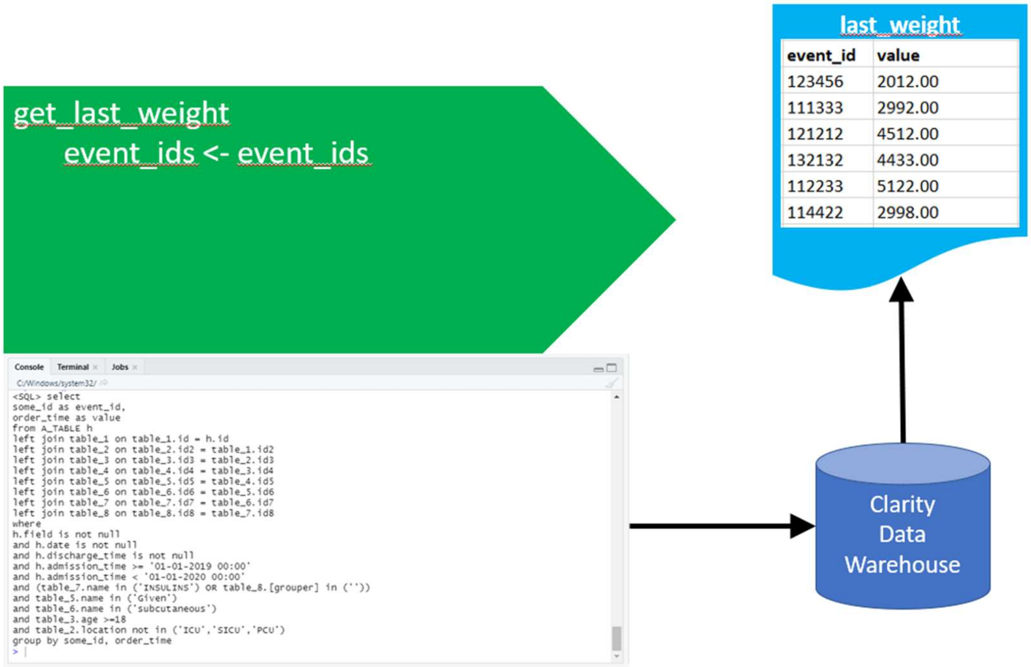
```
<SQL> select
some_id as event_id,
order_time as value
from A_TABLE h
left join table_1 on table_1.id = h.id
left join table_2 on table_2.id2 = table_1.id2
left join table_3 on table_3.id3 = table_2.id3
left join table_4 on table_4.id4 = table_3.id4
left join table_5 on table_5.id5 = table_4.id5
left join table_6 on table_6.id6 = table_5.id6
left join table_7 on table_7.id7 = table_6.id7
left join table_8 on table_8.id8 = table_7.id8
where
h.field is not null
and h.date is not null
and h.discharge_time is not null
and h.admission_time >= '01-01-2019 00:00'
and h.admission_time < '01-01-2020 00:00'
and (table_7.name in ('INSULINS') OR table_8.[grouper] in (''))
and table_5.name in ('Given')
and table_6.name in ('subcutaneous')
and table_3.age >=18
and table_2.location not in ('ICU','SICU','PCU')
group by some_id, order_time
> |
```

event_id	value
123456	1/1/19 3:33 PM
111333	1/2/19 4:44 PM
121212	1/2/19 5:54 PM
132132	1/5/19 3:22 AM
112233	1/7/19 8:22 AM
114422	1/10/19 8:25 PM

Figure 5. The cohort creation module. This module uses the get_event() function to generate a SQL query which returns a list of medication orders.

3.6. Cohort Creation Module

First, to generate a cohort of all the medication orders which will be used to train and test her model, Dr. Green uses the get_event() function. This function takes the start and end time of the inclusion period, a pharmacologic class name or grouper identification number (an internal identifier within Epic which refers to a value set) to specify which medication orders should be included, a minimum age value which allows the user to exclude pediatric patients if necessary, and excluded locations. Once Dr. Green runs this function, a SQL query is automatically generated which connects to the Clarity Data Warehouse and extracts a list of event IDs, which are medication order numbers, and a corresponding list of values, which are the timestamps for each medication order (Figure 5). In addition, there is an option to specify the medication routes to include when retrieving medication orders.



```
<SQL> select
some_id as event_id,
order_time as value
from A_TABLE h
left join table_1 on table_1.id = h.id
left join table_2 on table_2.id2 = table_1.id2
left join table_3 on table_3.id3 = table_2.id3
left join table_4 on table_4.id4 = table_3.id4
left join table_5 on table_5.id5 = table_4.id5
left join table_6 on table_6.id6 = table_5.id6
left join table_7 on table_7.id7 = table_6.id7
left join table_8 on table_8.id8 = table_7.id8
where
h.field is not null
and h.date is not null
and h.discharge_time is not null
and h.admission_time >= '01-01-2019 00:00'
and h.admission_time < '01-01-2020 00:00'
and (table_7.name in ('INSULINS') OR table_8.[grouper] in (''))
and table_5.name in ('Given')
and table_6.name in ('subcutaneous')
and table_3.age >=18
and table_2.location not in ('ICU','SICU','PCU')
group by some_id, order_time
>
```

last_weight

event_id	value
123456	2012.00
111333	2992.00
121212	4512.00
132132	4433.00
112233	5122.00
114422	2998.00

Figure 6. Example of a feature extraction function. The get_last_weight() function returns a list of weights (in ounces) for each event ID (medication order ID) that it is supplied.

3.7. Feature Extraction Module

Next, Dr. Green uses the feature extraction module to retrieve data about the patient's last weight and last glucose prior to the new medication order. She uses the get_last_weight() function, and only needs to supply the list of event IDs she retrieved in the last module, which specify her cohort of medication orders. This function then generates a SQL query which is automatically sent to Clarity to retrieve a list of weights corresponding to each event ID (Figure 6).

Dr. Green next uses the get_last_lab() function to retrieve the glucose level corresponding to each event ID. To do this, she must pass the function both the event IDs and a list of laboratory test IDs (component IDs, which are an internal identification number within Epic) that specify glucose. These laboratory test IDs are easy to find within the Epic system when reviewing laboratory results. Notably, the laboratory functions in MOSS are written to automatically parse any test values that are returned from text to numeric values. This is useful in the frequent case where a laboratory value may be returned as e.g., '>600' or '<25', signifying that the laboratory result is at the extreme limit of the assay. Such results are converted to numeric values (e.g., 600 and 25) automatically by MOSS so that they can be more easily included in predictive models.

Finally, Dr. Green uses the get_next_lowest_lab() function to retrieve data for her outcome of interest, which is whether the patient experiences a low blood sugar after the insulin order. To use this function, she supplies the event IDs which specify her cohort of medication orders, the laboratory test IDs to specify that glucose is the laboratory test of interest, and a 'lookahead' of 24 to specify that she wants to look for glucose results which resulted in the 24 hours following the insulin order.

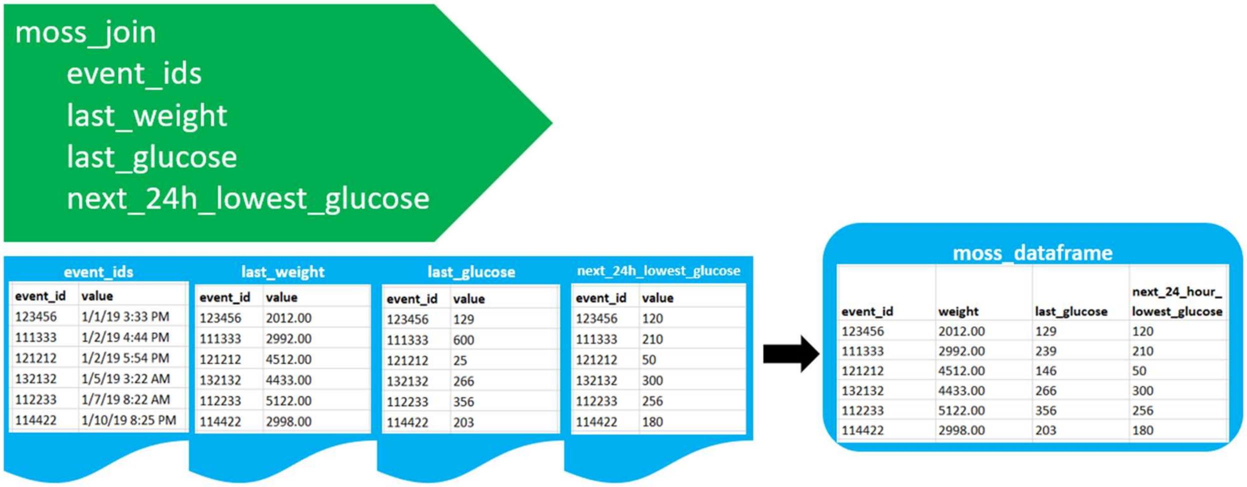
MOSS contains a suite of feature extraction functions, including functions for extracting patient characteristics, diagnoses, vital signs, laboratory results, medications, and details about the new medication order, such as the dose, unit, frequency, and route (Table 1).

Patient Characteristics and Diagnoses	**Vital Signs and Laboratory Results**	**Medications**	**Characteristics of Medication Order**
get_age()	get_flowsheet_row()	get_insulin_order_score()	get_med_name()
get_sex()	get_last_sbp()	get_insulin_infusion_total()	get_thera_class_name()
get_last_weight()	get_min_sbp()	get_insulin_admin_total()	get_pharm_class_name()
get_adm_diag_binary()	get_last_dbp()	get_med_score()	get_pharm_subclass_name()
get_hosp_prob_binary()	get_lab()	get_med_admin_total()	get_simple_generic_name()
get_problem_list_binary()	get_last_lab()	get_med_infusion_total()	get_med_dose()
get_problem_list_icd10()	get_last_lowest_lab()	get_med_admin_binary()	get_med_dose_range_min()
get_hosp_problem_list_icd10()	get_next_lab()	get_med_admin_count()	get_med_dose_range_max()
get_all_diag_binary()	get_next_lowest_lab()	get_opioid_admin_total()	get_med_unit()
get_some_diag_binary()			get_med_frequency()
			get_med_route()
			get_PO_med()

Table 1. Examples of MOSS feature extraction functions.

3.8. Dataset Creation

Once Dr. Green has retrieved her cohort and features, she uses the dataset creation module to join all of her data into a single dataset. This dataset can be used by the MOSS testing and evaluation modules, but is also a standard format which can be used by any of the multitude of predictive modeling packages in R. Finally, she can use the train_test_split() function to split her dataset randomly into a training set and a test set.



event_ids

event_id	value
123456	1/1/19 3:33 PM
111333	1/2/19 4:44 PM
121212	1/2/19 5:54 PM
132132	1/5/19 3:22 AM
112233	1/7/19 8:22 AM
114422	1/10/19 8:25 PM

last_weight

event_id	value
123456	2012.00
111333	2992.00
121212	4512.00
132132	4433.00
112233	5122.00
114422	2998.00

last_glucose

event_id	value
123456	129
111333	600
121212	25
132132	266
112233	356
114422	203

next_24h_lowest_glucose

event_id	value
123456	120
111333	210
121212	50
132132	300
112233	256
114422	180

moss_dataframe

event_id	weight	last_glucose	next_24_hour_lowest_glucose
123456	2012.00	129	120
111333	2992.00	239	210
121212	4512.00	146	50
132132	4433.00	266	300
112233	5122.00	356	256
114422	2998.00	203	180

Figure 7. The dataset creation module. Event IDs and features are joined into a single dataset which can be used by modeling functions.

3.9. Model Training Module

Dr. Green uses the model training module to train her predictive model. To use the moss_train() function, she must supply her dataset, whether she would like to remove null values (this is recommended, as most algorithms cannot accept null values), and the algorithm she would like to use. The moss_train() function is a user-friendly function that supports logistic regression, regularized logistic regression, random forest, XGBoost, and K nearest neighbors by creating a wrapper around functions from the 'caret' package created by Max Kuhn[44]. Some benefits and limitations of these algorithms are presented in Table 2. More advanced R users may prefer to use the MOSS dataset with the 'caret' package directly, or another R modeling package.

Model	**Strengths**	**Limitations**
Logistic regression	Fast to train, efficient, interpretable, can implement in Epic	Theoretically less powerful than other algorithms
Regularized logistic regression	Helps to avoid over-fitting with high dimensional datasets	Slow to train
Random forest	Unlikely to overfit, can implement in Epic	Slow to train, black box
XGBoost	Fast, unlikely to overfit	Black box
K Nearest Neighbors	Clinically intuitive	Slow to train, needs balanced dataset

Table 2. Strengths and limitations of selection of algorithms used for machine learning.

3.10. Model Evaluation Module

Finally, Dr. Green evaluates her model with the MOSS model evaluation model, using the moss_evaluate() function, which takes the model trained in the previous module, as well as the dataset to test the module with (typically, the test set) and generates a receiver operator characteristic (ROC) curve and area under the curve (AUC). She is also able to generate precision and recall results for her model, which help her plan for future integration of her model as a machine learning-based medication alert.

3.11. Diagnoses in MOSS

MOSS' suite of feature extraction functions (Table 1) includes a number of functions to retrieve patient diagnosis from the admission diagnosis list, hospital problem list, and problem list. These diagnosis functions fall into two groups: 1) 'regex' functions which use regular expressions to extract diagnoses as a binary feature and 2) 'CCSR' functions which use diagnosis categories to rather than regular expressions to extract diagnoses. The 'CCSR' functions make the Agency for Healthcare Research and Quality's Clinical Classifications Software Refined[45] (CCSR) which group ICD-9 and ICD-10 diagnoses into clinically meaningful categories. While 'regex' diagnosis extraction functions require users to determine themselves which diagnoses to focus on, and how to capture them (such as using the expressions 'CHF' and 'heart failure' to look for cases of congestive heart failure), the 'CCSR' diagnosis extraction functions allow the user to use all CCSR categories, if desired, and does not require the user to come up with any regular expressions in order to extract diagnoses.

3.12. Medication scores

Medications in EHRs are represented by prescription records which must be transformed before they can be incorporated into prediction models. For example, having several records of different antihypertensive prescriptions with different start and end dates is less useful than a representation of the total amount of antihypertensive medication a patient is on. Similarly, knowing that a patient has a new antihypertensive order is less useful than knowing whether this is an increase or decrease relative to her previous medication regimen. To represent medication amounts in MOSS, we created a set of functions that calculate a cumulative medication score, with support for antihypertensives, insulin, and opioid medications. These functions are described below.

3.13. Cumulative Antihypertensive Score

The therapeutic intensity score has previously been studied as a summary measure which can be used to represent the total number and dose of antihypertensive medications in a way that indicates the expected blood pressure lowering effect of therapy in outpatients[46]. This score is calculated by adding up the percentage of maximum of each prescribed antihypertensive a patient is on. For example, if a patient is on 20 mg lisinopril, which has a maximum dose of 40 mg daily, and 50 mg of hydrochlorothiazide, which has a maximum dose of 50 mg, he or she would have a therapeutic intensity score as calculated below.

$$Therapeutic\ intensity\ score = \frac{20mg}{40mg} + \frac{50mg}{50mg} = 1.5$$

To provide support within MOSS for predictive modeling around antihypertensives, we developed a similar method. However, rather than using the maximum daily dose, we mined training data to determine the median daily dose. First, we grouped antihypertensives at the generic level by intravenous (IV) vs. oral (PO) administration route. We then determined the median daily dose within the training set. Finally, we determined the average change in blood pressure in the 24 hours following an antihypertensive order by subtracting the minimum blood pressure found in the 24 hours following the antihypertensive order from the last blood pressure measured prior to the antihypertensive order. The cumulative antihypertensive score was calculated for each patient by determining each patient's 24-hour antihypertensive dose as a proportion of the maximum dose for that generic medication and route, multiplied by the average change in blood pressure.

To further illustrate how the cumulative antihypertensive score is calculated, let us take the example of a patient on 20 mg lisinopril and 50 mg hydrochlorothiazide. Suppose that the median daily dose of lisinopril within the training set is found to be 10 mg, and the average maximum drop in the systolic blood pressure is 43 mmHg. In the same way, suppose that the median daily dose of PO hydrochlorothiazide is 25 mg and the average maximum drop in the systolic blood pressure is 40 mm Hg. The calculation for this patient's cumulative antihypertensive score is shown below.

$$Cumulative\ antihypertensive\ score = \frac{20mg}{10mg} * 43 + \frac{50mg}{25mg} * 40 = 166$$

MOSS contains a mapping table with the mined median daily doses and average drop in systolic blood pressures that is used in the antihypertensive score functions.

3.14. Cumulative Insulin Score

Insulin is measured in International Units (units), which is standardized across different formulations of insulin. For this reason, totaling up the combined effect of different insulin prescriptions is relatively straightforward. The get_insulin_order_score() and get_insulin_admin_score() functions calculate the cumulative insulin score by totaling the amount of all insulin units a patient was ordered for, or was administered, over a 24-hour period. Similarly, MOSS contains functions for extracting insulin infusions, by totaling up the amount of insulin a patient was scheduled to receive, or did receive, over a 24-hour period based on the rate of the infusion and the amount of time the order was active.

3.15. Cumulative Opioid Score

MOSS also supports predictive modeling around medication orders for opioids. Because different opioid medications have different potencies, doses must be converted to morphine milligram equivalents (MMEs) in order to calculate a standardized daily dose. We used the Center for Disease Control and Prevention's MME conversion table to create a mapping table[47]. This mapping table is used by MOSS opioid functions to calculate the MME daily dose.

3.16. Connection to Clarity

Central to MOSS' architecture is the ability to connect to the Clarity data warehouse from within R. To accomplish this, MOSS makes use of the dbConnect function in R, which creates a connection using the necessary authentication mechanisms[48]. For a MOSS user to access Clarity, he or she must be logged into a computer with the correct credentials on the appropriate network. Once this connection is created, it is then passed as an object to MOSS' feature extraction functions and used to send SQL queries to Clarity.

3.17. Conclusion

MOSS is designed to make the process of training clinical prediction modules for medication safety approachable for users who may not have advanced SQL knowledge, or understanding of the Clarity data model. Next, we will look at the results of using MOSS for two clinical scenarios: predicting low blood pressure, or ‘hypotension’ after prescription of a blood pressure lowering medication (‘antihypertensive’) and predicting low blood sugar (‘hypoglycemia’) after prescription of insulin.

CHAPTER 4

MOSS Use Case #1: Predicting Antihypertensive-Induced Hypotension

4.1. Introduction

In this chapter, we discuss a proof-of-concept example where MOSS is used to train a model to predict hypotension in patients who are prescribed an antihypertensive medication, a real-life adverse drug event which can cause patient harm.

4.2. Background

Hypotension is a frequent and life-threatening condition in hospitalized patients and may often be caused by medical treatment. Patients exposed to hypotension are at increased risk for organ damage related to insufficient blood flow, such as myocardial injury and kidney injury. Iatrogenic hypotension was found to be the third most frequent cause of medical errors in the hospital by one research group[49]. Blood pressure medications are on the Institute for Safe Medication Practices' list of high-risk medications[50].

Most studies of inpatient hypotension have focused on patients undergoing anesthesia and intubation, especially in intensive care units (ICUs). One group developed a prediction model for hypotension following endotracheal intubation in medical and surgical ICUs[51]. They reported a C-statistic of 0.71 to 0.75 for their logistic regression, depending on inclusion criteria. Another group used multiple machine-learning classification techniques to predict hypotension during induction of general anesthesia and found AUCs from 0.63 (support vector machines) to 0.76 (gradient boosting machine), with an AUC of 0.71 for logistic regression[52]. In a similar published report of using different machine learning methods to predict post-induction hypotension, the AUC was 0.84 for random-forest, 0.78 for Naïve Bayes, 0.76 for logistic regression, and 0.76 for an artificial-neural-network[53]. Models for predicting hypotension during hemodialysis have also been reported, such as one study which used a deep neural network to predict intradialytic hypotension, with an accuracy of 64.97%[54].

No study that we know of has built a predictive model around iatrogenic hypotension in patients on the non-ICU medical wards of a hospital. Here, we use MOSS to build a predictive model for antihypertensive-induced hypotension in non-ICU inpatient adults.

4.3. Materials and methods

4.3.1. Study site and population

This study was conducted at Vanderbilt University Medical Center (VUMC), a large urban academic medical center in Nashville, Tennessee. VUMC's EHR, Epic, was implemented in November 2017 and includes a data warehouse, Clarity, which houses clinical data including medications and laboratory results (© 2021 Epic Systems Corporation). We included in our study population all patients aged 18 or older who had orders placed during an inpatient admission between January 1, 2018 and December 31, 2019. We excluded patients in intensive care units or palliative care units. This study was approved by the VUMC institutional review board.

4.3.2. Definition of Outcome

We defined antihypertensive-induced hypotension as a low systolic blood pressure (<90 mmHg) within 24 hours after ordering an antihypertensive.

4.3.3. Features

We used MOSS' feature extraction module to retrieve features for use in our hypotension model. Features used are listed in Table 3. We made use of MOSS' cumulative antihypertensive score functions to determine antihypertensive scores for patients at 24 hours and 1 minute prior to the new medication order and including the new medication order. Data retrieved were joined into a dataset using the MOSS dataset creation module.

Patient characteristics
Age
Sex
Weight
Diagnoses
Diagnoses on problem list, hospital problem list, and admission diagnoses (e.g., congestive heart failure, atrial fibrillation, hypertension, hypotension, end stage renal disease)
Vitals
Systolic blood pressure
Diastolic blood pressure
Laboratory values
GFR <60
WBC <4
WBC >14
Lactate >2
Medications
Antihypertensives on home medication list
Antihypertensives ordered and administered

Table 3. Features included in MOSS hypotension model.

4.3.4. Diagnosis Categories

We compared frequencies of CCSR diagnosis categories in positive and negative cases and incorporated as features only those CCSR categories with at least 5% absolute difference in frequency between positive and negative cases, as preliminary data had shown worse performance when using all CCSR categories.

4.3.5. Missing Data

Medication orders with missing data were excluded using case-wise deletion, as preliminary data showed that less than 5% of medication orders were missing data, and these missing data were primarily blood pressure measurements. We determined that any future clinical decision support predicting hypotension was unlikely to be useful in patients for whom blood pressure data was not available.

4.3.6. Model Development and Evaluation

We randomly split data into a training set (80%) and test set (20%). Models were trained with the training set and tested both on the training set and the test set. We performed a 10-fold cross-validation within the training set. We compared the performance of logistic regression, random forest, XGBoost, and K Nearest Neighbors. For each model, we graphed a receiver operating characteristic (ROC) curve and determined the area under the curve (AUC). We also looked at the precision of each model for a recall of approximately 50%. We determined the top 10 most significant predictors, ranked by the absolute value of the z-score. Models were run on the Vanderbilt Advanced Computing Center for Research and Education (ACCRE, www.vanderbilt.edu/accre).

4.4. Results

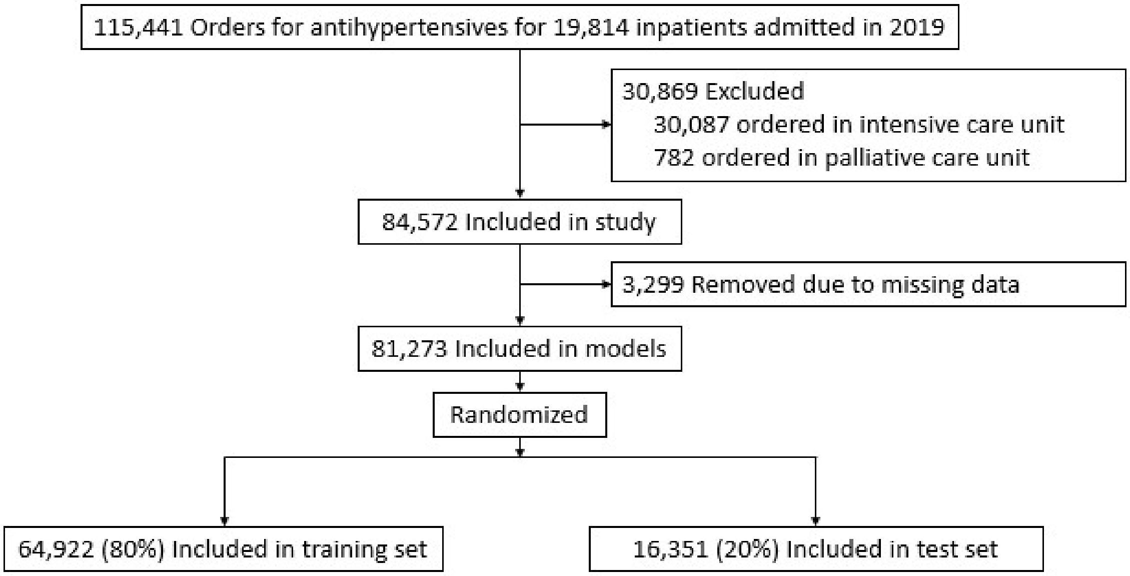

115,441 Orders for antihypertensives for 19,814 inpatients admitted in 2019
30,869 Excluded
30,087 ordered in intensive care unit
782 ordered in palliative care unit
84,572 Included in study
3,299 Removed due to missing data
81,273 Included in models
Randomized
64,922 (80%) Included in training set
16,351 (20%) Included in test set

Figure 8. Flow diagram for hypotension model.

A flow diagram is presented in Figure 8. Of 115,441 orders for antihypertensives retrieved during the study period, 30,869 (27%) were excluded, and an additional 3,299 (3%) were removed due to missing data, leaving 81,273 orders used to build and test predictive models. Of these 81,273 orders, 8,039 (10%)

were followed by hypotension within the following 24 hours (Table 4). Selected baseline characteristics for patients at the time of the antihypertensive order are listed in Table 4.

Predictor	
Age, mean (SD)	60 (17)
Sex, N (%)	
Female	36,365 (45)
Male	44,908 (55)
Antihypertensives on home med list	
None	22,494 (28)
One	11,328 (14)
2+	47,451 (58)
Diagnoses	
Hypertension	27,630 (34)
Congestive heart failure	22,464 (28)
Atrial fibrillation	9,440 (12)
End stage renal disease	4,326 (5)
Hypotension	1,476 (2)
Laboratory values	
Last GFR < 60, N (%)	31,293 (39)
Last WBC < 4, N (%)	4,648 (6)
Last WBC > 14, N (%)	8,981 (11)
Last lactate >2, N (%)	65 (1)
Blood pressure	
Last SBP, mean (SD)	135 (26)
Last DBP, mean (SD)	77 (17)
Lowest SBP in last 24 hours, mean (SD)	119 (23)
Outcome	
Hypotension, N (%)	8,039 (10)

Table 4. Baseline characteristics and outcome, hypotension model.

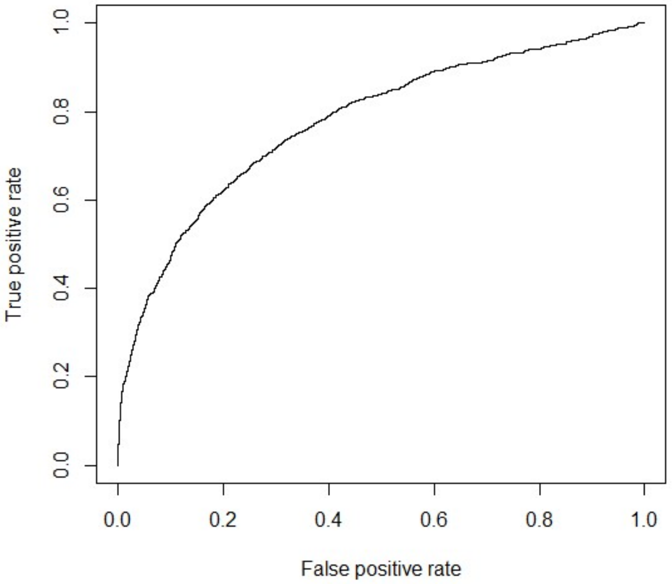


Figure 9. Area under the curve for hypotension model using logistic regression.

Model	Logistic Regression	Random Forest	XGBoost	K Nearest Neighbors
Hypotension	0.78	0.77	0.79	0.64

Table 5. AUCs for hypotension model comparing logistic regression, random forest, XGBoost, and K Nearest Neighbors.

4.4.1. Model Performance

The AUC for the hypotension model using logistic regression was 0.80 when tested on the training set, unchanged with 10-fold cross-validation, and 0.78 when tested on the test set (**Error! Reference source not found.**, Table 5). Both random forest and XGBoost performed similarly, with AUCs of 0.77 and 0.79 respectively. K Nearest Neighbors was worse with an AUC of 0.64. We determined that using the logistic regression model, a precision of 0.31 corresponded to a recall of 0.51. The most significant predictors for the hypotension model were the patient's lowest systolic blood pressure in the 24 hours prior to ordering the new medication, the last systolic blood pressure, and the patient's weight (Figure 10).

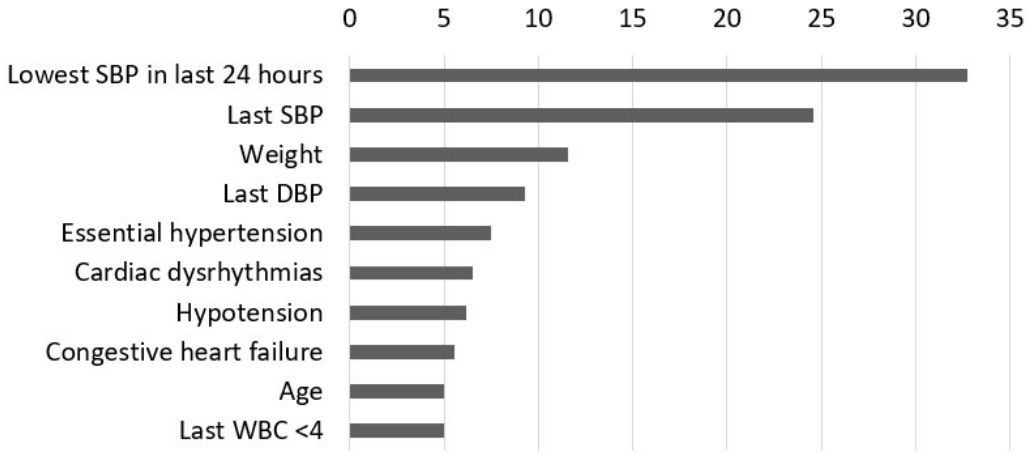


Figure 10. Feature importance for hypotension model. The top 10 most significant features for the hypotension model are displayed in a bar chart, ranked by the absolute value of the z-score.

4.5. Discussion

Using MOSS, we generated a clinical prediction model for hypotension in non-ICU inpatients with an AUC similar to those reported for post-intubation and post-induction hypotension. While a precision of 0.31 and recall of 0.51 leave room for improvement, they are higher than the precision and recall of typical table-based alerts.

The most important features for this model were the patient's lowest systolic blood pressure in the last 24 hours and the last systolic blood pressure recorded before the new order for an antihypertensive medication. This is a similar finding to another study which tried to predict hypotension after induction with machine learning, where the patient's lowest systolic blood pressure, lowest mean blood pressure, and mean systolic blood pressure before intubation were the most important factors[53].

XGBoost had the best performance, but the AUC for logistic regression was also acceptable. Given the proposed application of this model within a medication alert, it would be preferable to use logistic regression due to better explainability of the prediction. For example, when displaying an alert warning about hypotension, the logistic regression model could include patient-specific factors, such as the most recent systolic blood pressure, that put the patient at risk for hypotension (Figure 2).

4.6. Conclusion

This chapter demonstrates the potential of MOSS to facilitate the development of a prediction model for antihypertensive-induced hypotension. Using MOSS' feature extraction functions, we were able to automatically generate SQL queries to extract data from the Clarity database. While the positive predictive value for our hypotension alert showed improvement over common table-based alerts, it may be desirable to implement this alert at a higher positive predictive value with lower sensitivity to prevent alert fatigue. Determining the best threshold for alerting would depend upon clinical priorities and further evaluation of the impact on such an alert on workflow.

CHAPTER 5

MOSS Use Case #2: Predicting Insulin-Induced Hypoglycemia

5.1. Introduction

In this chapter, we discuss a second proof-of-concept example for the use of MOSS to develop a clinical prediction model. This time, we use MOSS to train a model predicting hypoglycemia in patients who are prescribed insulin. Hypoglycemia is a common condition in hospitalized patients, especially those with diabetes. It can be life-threatening, and for this reason, clinical prediction models for hypoglycemia have been an area of interest in recent years.

5.2. Background

Reduction in average glucose levels is a key target of therapy for diabetes, since it can prevent microvascular complications such as retinopathy, nephropathy, and neuropathy, and may prevent negative outcomes such as myocardial infarction[55]. However, therapy to lower blood glucose can also put patients with diabetes at risk for hypoglycemia, a condition which manifests as autonomic symptoms such as tremor, anxiety, and sweating, as well as symptoms due to decreased glucose availability to the brain, such as dizziness and confusion. In some cases, hypoglycemia can lead to coma and death. Insulin is on the Institute for Safe Medication Practices' list of high-risk medications[50]. Recently, models for insulin-associated hypoglycemia have been published, with one study reporting a C-statistic of 0.86[56], and another study reporting an AUC of 0.75 with logistic regression vs. 0.96 with XGBoost[57]. Here, we use MOSS to build a predictive model for insulin-induced hypoglycemia in non-ICU inpatient adults.

5.3. Materials and methods

5.3.1. Study site and population

This study was conducted at VUMC, a large urban academic medical center in Nashville, Tennessee. VUMC's EHR, Epic, was implemented in November 2017 and includes a data warehouse, Clarity, which houses clinical data including medications and laboratory results (© 2021 Epic Systems Corporation). We included in our study population all patients aged 18 or older who had orders placed during an inpatient admission between January 1, 2019 and December 31, 2019. We excluded patients in intensive care units or palliative care units. This study was approved by the VUMC institutional review board.

5.3.2. Definition of Outcome

We defined insulin-induced hypoglycemia, as a low blood glucose (<70 mg/dL) within 24 hours after ordering insulin.

5.3.3. Features

We used MOSS' feature extraction module to retrieve features for use in our hypoglycemia model. Features used are listed in Table 6. We made use of MOSS' insulin score functions to determine insulin scores for patients at 24 hours and 1 minute prior to the new medication order and including the new medication order. Data retrieved were joined into a dataset using the MOSS dataset creation module.

Patient characteristics
Age
Sex
Weight
Diagnoses
Diagnoses on problem list, hospital problem list, and admission diagnoses (e.g., diabetes, type 1 diabetes)
Laboratory values
Glucose (last glucose, lowest glucose in last 24 hours)
GFR <60
WBC <4
WBC >14
Lactate >2
Medications
Insulins on home medication list
Insulins ordered and administered
Dose, route, and units of new insulin order
Diet
Whether last diet order was nil per os

Table 6. Features included in MOSS Hypoglycemia Model.

5.3.4. Diagnosis Categories

We compared frequencies of CCSR diagnosis categories in positive and negative cases and incorporated as features only those CCSR categories with at least 5% absolute difference in frequency between positive and negative cases, as preliminary data had shown worse performance when using all CCSR categories.

5.3.5. Missing Data

Medication orders with missing data were excluded using case-wise deletion, as preliminary data showed that less than 5% of medication orders were missing data, and these missing data were primarily blood glucose measurements. We determined that any future clinical decision support predicting hypoglycemia was unlikely to be useful in patients for whom blood glucose data was not available.

5.3.6. Model Development and Evaluation

We randomly split data into a training set (80%) and test set (20%). Models were trained with the training set and tested both on the training set and the test set. We performed a 10-fold cross-validation within the training set. We compared the performance of logistic regression, random forest, XGBoost, and K Nearest Neighbors. For each model, we graphed a receiver operating characteristic (ROC) curve and determined the area under the curve (AUC). We also looked at the precision of each model for a recall of approximately 50%. We determined the top 10 most significant predictors, ranked by the absolute value of the z-score. Models were run on the Vanderbilt Advanced Computing Center for Research and Education (ACCRE, www.vanderbilt.edu/accre).

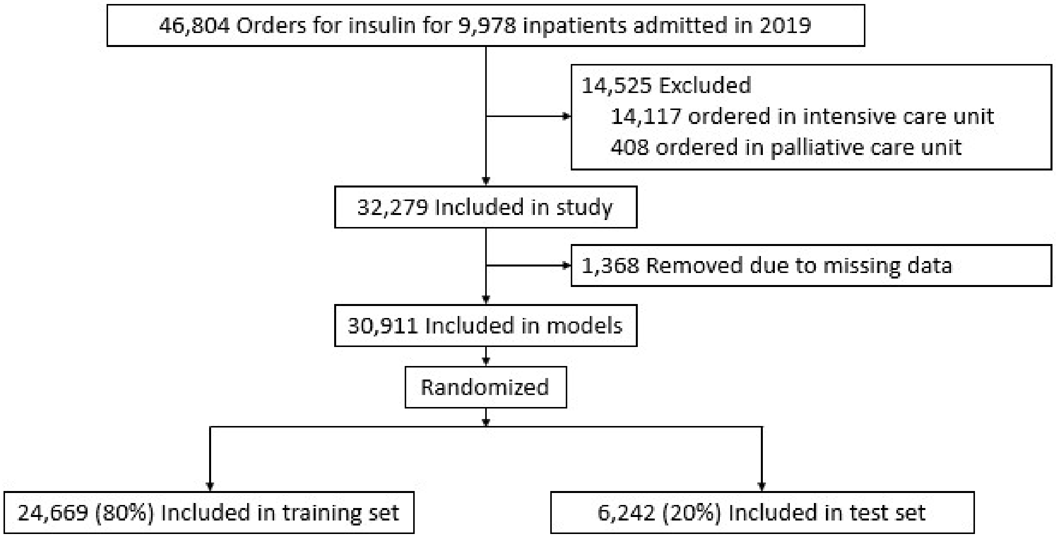

46,804 Orders for insulin for 9,978 inpatients admitted in 2019
14,525 Excluded
14,117 ordered in intensive care unit
408 ordered in palliative care unit
32,279 Included in study
1,368 Removed due to missing data
30,911 Included in models
Randomized
24,669 (80%) Included in training set
6,242 (20%) Included in test set

Figure 11. Flow diagram for hypoglycemia model.

5.4. Results

A flow diagram is presented in Figure 11. Of 46,804 orders for insulin retrieved during the study period, 14,525 (31%) were excluded, and an additional 1,368 (3%) were removed due to missing data, leaving 30,911 orders used to build and test predictive models. Of these 30,911 orders, 2,599 (8%) were followed by hypoglycemia within the following 24 hours (Table 7). Selected baseline characteristics for patients at the time of the insulin order are listed in Table 7.

Predictors	
Age, mean (SD)	58 (16)
Sex, N (%)	
Female	13,350 (43)
Male	17,561 (57)
Insulins on home med list	
None	13,315 (43)
One	5,224 (17)
2+	12, 372 (40)
Diagnoses	
Diabetes	21,138 (68)
Type 1 Diabetes	1,468 (5)
End stage renal disease	2,359 (8)
Laboratory values	
Last GFR < 60, N (%)	14,639 (47)
Last WBC < 4, N (%)	2,107 (7)
Last WBC > 14, N (%)	4,226 (14)
Last lactate >2, N (%)	577 (2)
Last glucose, mean (SD)	219 (109)
Lowest glucose in last 24 hours, mean (SD)	170 (92)
Diet nil per os	5,850 (19)
Outcome	
Hypoglycemia, N (%)	2,599 (8)

Table 7. Baseline characteristics and outcome, hypoglycemia model.

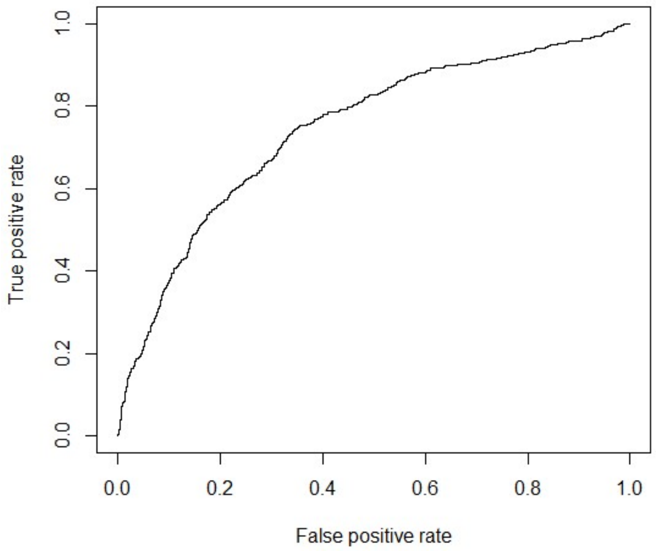


Figure 12. Area under the curve for hypoglycemia model using logistic regression.

Model	Logistic Regression	Random Forest	XGBoost	K Nearest Neighbors
Hypoglycemia	0.74	0.78	0.76	0.62

Table 8. AUCs for hypoglycemia model comparing logistic regression, random forest, XGBoost, and K Nearest Neighbors.

5.4.1. Model Performance

The AUC for the hypoglycemia model using logistic regression was 0.74 when tested on the training set, unchanged with 10-fold cross-validation, and 0.74 when tested on the test set (Figure 12, Table 8). Both random forest and XGBoost performed slightly better, with AUCs of 0.77 and 0.76 respectively. K Nearest Neighbors was worse with an AUC of 0.62. We determined that using the logistic regression model, a precision of 0.22 corresponded to a recall of 0.51. The most significant predictors for the hypoglycemia model were the patient's lowest blood glucose in the 24 hours prior to ordering the new medication, the medication route, and the patient's weight.

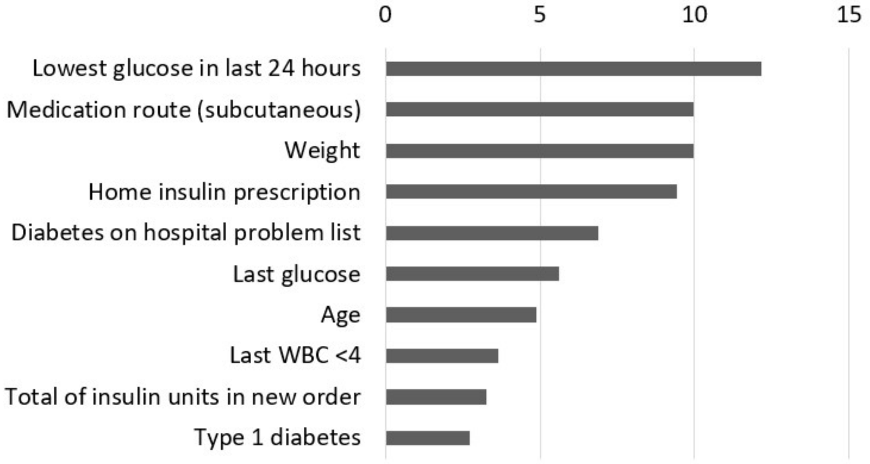


Figure 13. Feature importance for hypoglycemia model. The top 10 most significant features for the hypoglycemia model are displayed in a bar chart, ranked by the absolute value of the z-score.

5.5. Discussion

Using MOSS, we were able to generate a clinical prediction model for inpatient hypoglycemia. The precision of 0.22 and recall of 0.51 leave room for improvement, but still outperform typical table-based alerts.

The model performed worse than another published model for hypoglycemia which reported an AUC of 0.96 using XGBoost[57]. They reported a number of significant predictors which we did not include in our model, such as diastolic blood pressure, oxygen saturation, temperature, albumin levels, sulfonylurea use, metformin use, and procedures. Many of these features are already supported with MOSS' feature extraction module, and present an opportunity for further improvement of the model.

The most important feature for this model was the patient's lowest blood glucose in the last 24 hours prior to the new order for an antihypertensive medication. This parallels our finding for the hypotension model, in which the patient's lowest systolic blood pressure in the last 24 hours was the most important feature.

For predicting hypoglycemia, random forest had the best performance, but the AUC for logistic regression was also acceptable. It would likely be preferable to use logistic regression given its better explainability, though both logistic regression and random forest can be implemented as prediction models in Epic according to our preliminary investigation.

5.6. Conclusion

This chapter demonstrates the potential of MOSS to facilitate the development of a prediction model for a second clinical scenario further illustrating the flexibility of MOSS in generating different prediction models. For this use case, XGBoost outperformed other algorithms. This use case highlights one of the benefits of MOSS: its support for experimenting with different models and features. A clinician without SQL knowledge or advanced machine learning experience would be able to use MOSS' functions to add new models or features without the time-consuming back-and-forth of working with a non-clinician data analyst.

CHAPTER 6

Discussion

6.1. Introduction

As EHRs become widespread, clinical data amasses, and machine learning methods advance, the need arises for a platform to catalyze the process of learning from data in healthcare to make care safer and better for patients.

6.2. The MOSS Architecture

As laid out in chapter 3, the MOSS architecture presents an example of how to standardize the process of generating clinical prediction models. Key themes of the MOSS architecture including streamlining the process of generating SQL queries, having modular components and flexibility among functions, and supporting the feature engineering process by preparing data for use in models.

6.3. Automatic Generation of SQL Queries

As we designed this platform, the objectives were to both make this process more approachable for individuals not familiar with SQL, as well as to make this process more efficient for users with advanced SQL knowledge. The ability of MOSS feature extraction functions to automatically build SQL queries saves users the time of rewriting SQL queries and remembering which tables and fields hold data of interest. Moreover, when functions are run, users have the option to inspect the queries that are generated, which can serve as a learning tool for SQL coding and the Clarity data model.

6.4. Modularity and Flexibility

The design of MOSS emphasizes modularity and flexibility at multiple levels. Each MOSS module can function independently, allowing the user to pick and choose which modules to use. Furthermore, the feature extraction module allows each feature to be extracted independently before being joined into one dataset. This has the potential downside of slower performance; rather than creating one large SQL query to extract all the features at once, MOSS sends out multiple sequential queries. Some of these queries are redundant: one might use the get_last_lowest_lab() function multiple times with different time intervals for the same laboratory test, e.g., glucose. Each time the function is run, a temporary table is created with all the glucose results for the cohort. We could have engineered MOSS to re-use these temporary tables rather than recreating it each time a function is run, but the initial strategy was to make each part of MOSS standalone, ensuring that one function does not depend on another function. Further optimization is possible, although as designed, the MOSS feature extraction module has typically taken less than thirty minutes to extract one year of data. Such performance would not meet the requirements of point-of-care model generation for CDS, but should prove sufficient for the use case where a model is trained and updated proactively, as the predictions themselves are quickly made once the model has been trained.

6.5. Alternatives to Medication Orders for Cohort

As currently implemented, MOSS is designed to predict adverse outcomes at the time of medication orders. However, by substituting a different cohort in the cohort creation module, MOSS can be adapted to focus on medication administrations, or theoretically any other timed event. This would allow MOSS to generate predictions at the time of medication administration, which could lead to CDS for nurses at the time that medications are given. Such a model might provide another layer of safety, for example in the case that a blood pressure lowering agent was ordered when a patient's blood pressure was normal, but a patient's blood pressure fell prior to the scheduled time of administration. Theoretically, MOSS could be used to predict the risk of an adverse event at any point in time, such as at chart opening or at the time a laboratory test results.

6.6. Support for Feature Extraction

The MOSS feature extraction functions are designed to minimize time-consuming tasks of data cleaning, such as converting text laboratory results to numeric values. These functions often take a time period as an argument, so that the user can specify whether they would like to look for lab results from 24 hours prior to the medication order, or only 6 hours before. This flexibility around time periods presents the potential to optimize these as a hyperparameter. MOSS also provides support for combining similar medications to generate a medication score. While this might not be necessary for models like neural networks, in our experience logistic regression models have benefited from using medication scores as features.

6.7. Explainability

We demonstrated the ability of MOSS to generate predictive models for two use cases. As has previously been shown, there was only a small difference in the performance of more complicated machine learning methods over logistic regression. Logistic regression has the benefit of being interpretable, as opposed to other "black box" methods such as random forest and XGBoost. Miller and Masarie write that the most important intellect brought to any CDS system is that of the clinician[58]. Medication alerts that provide useful information to the clinician, such as through presentation of patient risk factors, will make the best use of the clinician's valuable intellect.

Despite the attractiveness of logistic regression, MOSS does allow the user to select other models. One major strength of MOSS is the ability to test and compare different algorithms once the MOSS data structure has been generated. Certain use cases, such as those that require a large feature space, may benefit from the use of more advanced machine learning algorithms. The decision to use such algorithms would have to weigh performance benefits and the downsides of using "black box" methods which make it hard to provide rationale when displaying an alert to users.

6.8. Limitations

While MOSS shows great potential as a platform for generating predictive models, further investigation is required to determine its effectiveness with models besides antihypertensive-induced hypotension and insulin-induced hypoglycemia. MOSS does not yet provide any support for unstructured data. Its regular expression tools provide a starting point, but adding more advanced natural language processing (NLP) methods has the potential to improve model performance. Finally, MOSS has not yet been integrated into the EHR, though preliminary investigation suggests this is possible in our hospital's EHR, especially with logistic regression and random forest models.

6.9. Future Work

6.9.1. Additional Use Cases

In our experience, compared to the time spent building out MOSS for our first use case, adding an additional use case was much faster, as it reused many of the MOSS functions that had already been constructed. It is our hope that with each additional use case that is added, MOSS becomes more flexible and adaptable for different purposes. With this in mind, we aim to add additional use cases in our future work, including opioid-induced respiratory depression, anticoagulant-induced bleeding, and nephrotoxic medication-induced acute kidney injury. Since MOSS is designed with the goal of preventing adverse events, we hope to add additional feature extraction modules that find potential safety events, such as significant events (rapid responses and codes), transfers to higher levels of care, and readmissions.

6.9.2. Unstructured Data

As we add additional use cases and improve existing models, we hope to use the large amount of unstructured data in the EHR, such as clinical notes. Parsing of natural language is a challenging task given the vastly large size of natural language and problems such as word ambiguity[59]. However, modern machine learning methods such as support vector machines, hidden Markov models, conditional random fields, and N-grams have been applied to NLP tasks and found to improve the performance of models. One study found that adding NLP to a clinical prediction model of outcomes among ICU patients improved the AUC from 0.899 to 0.922[60]. More recently, NLP techniques such as the Bidirectional Encoder Representations from Transformers (BERT), developed by Google, has shown promise for clinical texts[61]. We aim to use such techniques to capture unstructured information about significant patient events that may not exist in any structured fields, such as patient deterioration and resuscitation.

6.9.3. Making MOSS Accessible to Others

In designing MOSS as a platform to facilitate the generation of clinical prediction models for CDS, our goal has been to make the process approachable and accessible to others. Next steps include developing

MOSS into a user-friendly R package and providing MOSS to other researchers and personnel at our institution.

6.9.4. Integration of MOSS into the EHR

By making predictive modeling more approachable for clinicians, MOSS may lead to more clinically-relevant models that are amenable to implementation as CDS. Due to the design of MOSS, which centers around the process of medication ordering, MOSS models are set up to be integrated as CDS which may alert users at the time of ordering a medication. In our next stage of work, we aim to build out the MOSS pipeline to facilitate integration into the EHR. Building of effective CDS is not a simple task. Studies evaluating the effectiveness of CDS to improve clinical care have often found mixed results[62,63]. In the "Ten Commandments" for CDS, Bates et al. discussed strategies for CDS, such as improving the speed of systems, bringing information to the clinician at the right time in the workflow, monitoring the use of systems, keeping knowledge bases used to generate CDS up to date[64]. One way to facilitate this process will be to add an alert design module within MOSS that helps the user determine how often an alert would fire per day at any given threshold. Additionally, the ease of use of MOSS for generating prediction models would hopefully translate to more frequent updating and recalibration of models.

6.10. Portability of MOSS

It is our hope that as we further optimize MOSS for generating machine learning-based CDS at our institution, we will also find ways to replicate MOSS outside of our institution. This will be especially possible at other Epic sites, although small differences in each Epic site's instantiation of Clarity may require some adjustments to the queries which underly many MOSS functions. Translating MOSS to a different vendor system would likely take additional work. The use of OMOP CDM may be another strategy for improving the portability of MOSS[43]. However, using mapping may further distance MOSS models from the underlying clinical data, potentially affecting the performance of CDS. Overall, while adapting each one of the many MOSS functions for use at another institution may require some work, we hope that the architecture and design of MOSS may prove useful to other institutions seeking to catalyze the process of generating machine learning-based CDS.

6.11. Conclusion

The MOSS pipeline demonstrates an effective platform for using EHR data to build predictive models around medication safety. As designed, MOSS is an example of a system that combines self-service query tools for researchers with machine learning tools that are modular and flexible. Such tools can be used to remove technical barriers and capitalize upon the vast number of clinicians and institutional personnel who are otherwise poised to make contributions to the grand challenge of learning from clinical data to improve CDS.

REFERENCES

1. Charles D, Gabriel M, Searcy T. Adoption of Electronic Health Record Systems among U.S. Non-Federal Acute Care Hospitals: 2008-2014. :10.

2. Porter ME, Lee TH. The Strategy That Will Fix Health Care. *Harv Bus Rev*. Published online October 1, 2013. Accessed March 12, 2021. https://hbr.org/2013/10/the-strategy-that-will-fix-health-care

3. The big-data revolution in US health care: Accelerating value and innovation | McKinsey. Accessed March 12, 2021. https://www.mckinsey.com/industries/healthcare-systems-and-services/our-insights/the-big-data-revolution-in-us-health-care

4. Arel I, Rose DC, Karnowski TP. Deep Machine Learning - A New Frontier in Artificial Intelligence Research [Research Frontier]. *IEEE Comput Intell Mag*. 2010;5(4):13-18. doi:10.1109/MCI.2010.938364

5. Makary MA, Daniel M. Medical error—the third leading cause of death in the US. *BMJ*. 2016;353:i2139. doi:10.1136/bmj.i2139

6. Meigs SL, Solomon M. Electronic Health Record Use a Bitter Pill for Many Physicians. *Perspect Health Inf Manag*. 2016;13(Winter). Accessed March 12, 2021. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4739443/

7. McCoy AB, Thomas EJ, Krousel-Wood M, Sittig DF. Clinical Decision Support Alert Appropriateness: A Review and Proposal for Improvement. *Ochsner J*. 2014;14(2):195-202.

8. Bryant AD, Fletcher GS, Payne TH. Drug Interaction Alert Override Rates in the Meaningful Use Era. *Appl Clin Inform*. 2014;5(3):802-813. doi:10.4338/ACI-2013-12-RA-0103

9. Dixon-Woods M, Amalberti R, Goodman S, Bergman B, Glasziou P. Problems and promises of innovation: why healthcare needs to rethink its love/hate relationship with the new. *BMJ Qual Saf*. 2011;20(Suppl 1):i47-i51. doi:10.1136/bmjqs.2010.046227

10. Davis SE, Lasko TA, Chen G, Siew ED, Matheny ME. Calibration drift in regression and machine learning models for acute kidney injury. *J Am Med Inform Assoc*. 2017;24(6):1052-1061. doi:10.1093/jamia/ocx030

11. Leape LL. Error in Medicine. *JAMA J Am Med Assoc*. 1994;272(23):1851. doi:10.1001/jama.1994.03520230061039

12. Bates DW, Spell N, Cullen DJ, et al. The costs of adverse drug events in hospitalized patients. Adverse Drug Events Prevention Study Group. *JAMA*. 1997;277(4):307-311.

13. Institute of Medicine (US) Committee on Quality of Health Care in America. *To Err Is Human: Building a Safer Health System*. (Kohn LT, Corrigan JM, Donaldson MS, eds.). National Academies Press (US); 2000. Accessed March 1, 2021. http://www.ncbi.nlm.nih.gov/books/NBK225182/

14. Medicine I of. *Preventing Medication Errors*.; 2006. doi:10.17226/11623

15. Bates DW. Effect of Computerized Physician Order Entry and a Team Intervention on Prevention of Serious Medication Errors. *JAMA*. 1998;280(15):1311. doi:10.1001/jama.280.15.1311

16. Sittig DF, Stead WW. Computer-based physician order entry: the state of the art. *J Am Med Inform Assoc*. 1994;1(2):108-123.

17. Mekhjian HS, Kumar RR, Kuehn L, et al. Immediate Benefits Realized Following Implementation of Physician Order Entry at an Academic Medical Center. *J Am Med Inform Assoc*. 2002;9(5):529-539. doi:10.1197/jamia.M1038

18. Kuperman GJ, Gibson RF. Computer Physician Order Entry: Benefits, Costs, and Issues. *Ann Intern Med*. 2003;139(1):31-39. doi:10.7326/0003-4819-139-1-200307010-00010

19. Duke JD, Li X, Dexter P. Adherence to drug—drug interaction alerts in high-risk patients: a trial of context-enhanced alerting. *J Am Med Inform Assoc*. 2013;20(3):494-498. doi:10.1136/amiajnl-2012-001073

20. Tcheng JE, National Academy of Medicine (U.S.), eds. *Optimizing Strategies for Clinical Decision Support: Summary of a Meeting Series*. National Academy of Medicine; 2017.

21. Jankovic I, Chen JH. Clinical Decision Support and Implications for the Clinician Burnout Crisis. *Yearb Med Inform*. 2020;29(1):145-154. doi:10.1055/s-0040-1701986

22. Sorita A, Bos JM, Morlan BW, Tarrell RF, Ackerman MJ, Caraballo PJ. Impact of clinical decision support preventing the use of QT-prolonging medications for patients at risk for torsade de pointes. *J Am Med Inform Assoc*. 2015;22(e1):e21-e27. doi:10.1136/amiajnl-2014-002896

23. Abrich VA, Ramakrishna H, Mehta A, Mookadam F, Srivathsan K. The possible role of propofol in drug-induced torsades de pointes: A real-world single-center analysis. *Int J Cardiol*. 2017;232:243-246. doi:10.1016/j.ijcard.2017.01.011

24. Goff David C., Lloyd-Jones Donald M., Bennett Glen, et al. 2013 ACC/AHA Guideline on the Assessment of Cardiovascular Risk. *Circulation*. 2014;129(25_suppl_2):S49-S73. doi:10.1161/01.cir.0000437741.48606.98

25. Prospective multicenter evaluation of the pulmonary embolism rule-out criteria - KLINE - 2008 - Journal of Thrombosis and Haemostasis - Wiley Online Library. Accessed March 22, 2021. https://onlinelibrary.wiley.com/doi/full/10.1111/j.1538-7836.2008.02944.x

26. Morse JM, Black C, Oberle K, Donahue P. A prospective study to identify the fall-prone patient. *Soc Sci Med 1982*. 1989;28(1):81-86. doi:10.1016/0277-9536(89)90309-2

27. Vallet-Pichard A, Mallet V, Nalpas B, et al. FIB-4: an inexpensive and accurate marker of fibrosis in HCV infection. comparison with liver biopsy and fibrotest. *Hepatol Baltim Md*. 2007;46(1):32-36. doi:10.1002/hep.21669

28. Christodoulou E, Ma J, Collins GS, Steyerberg EW, Verbakel JY, Van Calster B. A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models. *J Clin Epidemiol*. 2019;110:12-22. doi:10.1016/j.jclinepi.2019.02.004

29. Petersen C, Smith J, Freimuth RR, et al. Recommendations for the safe, effective use of adaptive CDS in the US healthcare system: an AMIA position paper. *J Am Med Inform Assoc*. 2021;28(4):677-684. doi:10.1093/jamia/ocaa319

30. Sawyer AM, Deal EN, Labelle AJ, et al. Implementation of a real-time computerized sepsis alert in nonintensive care unit patients*. *Crit Care Med*. 2011;39(3):469-473. doi:10.1097/CCM.0b013e318205df85

31. Gallagher D, Zhao C, Brucker A, et al. Implementation and Continuous Monitoring of an Electronic Health Record Embedded Readmissions Clinical Decision Support Tool. *J Pers Med*. 2020;10(3):103. doi:10.3390/jpm10030103

32. Stevens JA, Smith ML, Parker EM, Jiang L, Floyd FD. Implementing a Clinically Based Fall Prevention Program. *Am J Lifestyle Med*. 2020;14(1):71-77. doi:10.1177/1559827617716085

33. Kansagara D, Englander H, Salanitro A, et al. Risk Prediction Models for Hospital Readmission: A Systematic Review. *JAMA*. 2011;306(15):1688-1698. doi:10.1001/jama.2011.1515

34. Sittig DF, Wright A, Osheroff JA, et al. Grand challenges in clinical decision support. *J Biomed Inform*. 2008;41(2):387-392. doi:10.1016/j.jbi.2007.09.003

35. Lee Y, Bang H, Kim DJ. How to Establish Clinical Prediction Models. *Endocrinol Metab*. 2016;31(1):38-44. doi:10.3803/EnM.2016.31.1.38

36. Wilcox A, Vawdrey D, Weng C, Velez M, Bakken S. Research Data Explorer: Lessons Learned in Design and Development of Context-based Cohort Definition and Selection. *AMIA Summits Transl Sci Proc*. 2015;2015:194-198.

37. Gerry S, Bonnici T, Birks J, et al. Early warning scores for detecting deterioration in adult hospital patients: systematic review and critical appraisal of methodology. *BMJ*. 2020;369:m1501. doi:10.1136/bmj.m1501

38. Futoma J, Simons M, Panch T, Doshi-Velez F, Celi LA. The myth of generalisability in clinical research and machine learning in health care. *Lancet Digit Health*. 2020;2(9):e489-e492. doi:10.1016/S2589-7500(20)30186-2

39. Davis SE, Greevy RA, Lasko TA, Walsh CG, Matheny ME. Detection of calibration drift in clinical prediction models to inform model updating. *J Biomed Inform*. 2020;112:103611. doi:10.1016/j.jbi.2020.103611

40. Murphy SN, Gainer V, Chueh HC. A Visual Interface Designed for Novice Users to find Research Patient Cohorts in a Large Biomedical Database. *AMIA Annu Symp Proc*. 2003;2003:489-493.

41. Murphy S, Wilcox A. Mission and Sustainability of Informatics for Integrating Biology and the Bedside (i2b2). *eGEMs*. 2014;2(2). doi:10.13063/2327-9214.1074

42. Harris PA, Swafford JA, Edwards TL, et al. StarBRITE: The Vanderbilt University Biomedical Research Integration, Translation and Education portal. *J Biomed Inform*. 2011;44(4):655-662. doi:10.1016/j.jbi.2011.01.014

43. Hripcsak G, Duke JD, Shah NH, et al. Observational Health Data Sciences and Informatics (OHDSI): Opportunities for Observational Researchers. *Stud Health Technol Inform*. 2015;216:574-578.

44. Kuhn M. Building Predictive Models in *R* Using the **caret** Package. *J Stat Softw*. 2008;28(5). doi:10.18637/jss.v028.i05

45. Clinical Classifications Software Refined (CCSR). Accessed March 1, 2021. https://www.hcup-us.ahrq.gov/toolssoftware/ccsr/ccs_refined.jsp

46. Levy PD, Willock RJ, Burla M, et al. Total antihypertensive therapeutic intensity score and its relationship to blood pressure reduction. *J Am Soc Hypertens JASH*. 2016;10(12):906-916. doi:10.1016/j.jash.2016.10.005

47. Calculating Total Daily Dose of Opioids For Safer Dosage. :2.

48. Create a connection to a DBMS — dbConnect. Accessed March 12, 2021. https://dbi.r-dbi.org/reference/dbconnect

49. David G, Gunnarsson CL, Waters HC, Horblyuk R, Kaplan HS. Economic Measurement of Medical Errors Using a Hospital Claims Database. *Value Health*. 2013;16(2):305-310. doi:10.1016/j.jval.2012.11.010

50. High-Alert Medications in Acute Care Settings. Institute For Safe Medication Practices. Accessed March 1, 2021. https://www.ismp.org/recommendations/high-alert-medications-acute-list

51. Smischney NJ, Kashyap R, Khanna AK, et al. Risk factors for and prediction of post-intubation hypotension in critically ill adults: A multicenter prospective cohort study. *PLOS ONE*. 2020;15(8):e0233852. doi:10.1371/journal.pone.0233852

52. Kendale S, Kulkarni P, Rosenberg AD, Wang J. Supervised Machine-learning Predictive Analytics for Prediction of Postinduction Hypotension. *Anesthesiology*. 2018;129(4):675-688. doi:10.1097/ALN.0000000000002374

53. Kang AR, Lee J, Jung W, et al. Development of a prediction model for hypotension after induction of anesthesia using machine learning. *PLOS ONE*. 2020;15(4):e0231172. doi:10.1371/journal.pone.0231172

54. Chen J, Wu K, Moi S, Chuang L, Yang C. Deep Learning for Intradialytic Hypotension Prediction in Hemodialysis Patients. *IEEE Access*. 2020;8:82382-82390. doi:10.1109/ACCESS.2020.2988993

55. Cryer PE, Davis SN, Shamoon H. Hypoglycemia in Diabetes. *Diabetes Care*. 2003;26(6):1902-1912. doi:10.2337/diacare.26.6.1902

56. Mathioudakis NN, Everett E, Routh S, et al. Development and validation of a prediction model for insulin-associated hypoglycemia in non-critically ill hospitalized adults. *BMJ Open Diabetes Res Care*. 2018;6(1):e000499. doi:10.1136/bmjdrc-2017-000499

57. Ruan Y, Bellot A, Moysova Z, et al. Predicting the Risk of Inpatient Hypoglycemia With Machine Learning Using Electronic Health Records. *Diabetes Care*. 2020;43(7):1504-1511. doi:10.2337/dc19-1743

58. The demise of the Greek Oracle model for medical diagnostic systems (editorial) - Search results - Pascal and Francis Bibliographic Databases. Accessed March 12, 2021. https://pascal-francis.inist.fr/vibad/index.php?action=getRecordDetail&idt=4423629

59. Nadkarni PM, Ohno-Machado L, Chapman WW. Natural language processing: an introduction. *J Am Med Inform Assoc*. 2011;18(5):544-551. doi:10.1136/amiajnl-2011-000464

60. Marafino BJ, Park M, Davies JM, et al. Validation of Prediction Models for Critical Care Outcomes Using Natural Language Processing of Electronic Health Record Data. *JAMA Netw Open*. 2018;1(8):e185097-e185097. doi:10.1001/jamanetworkopen.2018.5097

61. Hao B, Zhu H, Paschalidis I. Enhancing Clinical BERT Embedding using a Biomedical Knowledge Base. In: *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics; 2020:657-661. doi:10.18653/v1/2020.coling-main.57

62. Hunt DL, Haynes RB, Hanna SE, Smith K. Effects of Computer-Based Clinical Decision Support Systems on Physician Performance and Patient OutcomesA Systematic Review. *JAMA*. 1998;280(15):1339-1346. doi:10.1001/jama.280.15.1339

63. Kaushal R, Shojania KG, Bates DW. Effects of Computerized Physician Order Entry and Clinical Decision Support Systems on Medication Safety: A Systematic Review. *Arch Intern Med*. 2003;163(12):1409-1416. doi:10.1001/archinte.163.12.1409

64. Bates DW, Kuperman GJ, Wang S, et al. Ten Commandments for Effective Clinical Decision Support: Making the Practice of Evidence-based Medicine a Reality. *J Am Med Inform Assoc*. 2003;10(6):523-530. doi:10.1197/jamia.M1370