Prediction and Privacy in Healthcare Analytics

Stacey Newman

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2016

Committee:

Martine De Cock

Anderson Nascimento

Shanu Sushmita

Ankur Teredesai

Program Authorized to Offer Degree:
Computer Science and Systems

University of Washington

Abstract

Prediction and Privacy in Healthcare Analytics

Stacey Newman

Chair of the Supervisory Committee:
Associate Professor Martine De Cock
Institute of Technology

In the past decade, the United States federal government has made improving the healthcare system a major focus with legislation such as the Patient Protection and Affordable Care Act and the financial incentives for meaningful use within the American Recovery and Reinvestment Act. This focus has caused an increase in data collection as part of Electronic Medical Record adoption by healthcare organizations, and with that data has come an increased effort towards improving quality of care while controlling growing costs. While data collection has increased significantly, understanding that data to produce predictive models that aid organizations with their quality and financial goals remains a challenge, especially in a complex domain with patient data privacy concerns. In this thesis we propose solutions in machine learning to address challenges in both quality of care and cost management as well as cryptographic protocols to allow development of machine learning models while protecting patient data. In the cases of our predictive models, we go as far as developing tools for medical providers, insurers, and patients to leverage our solutions. We discuss in detail the impact machine learning techniques can have on challenges in the healthcare domain and how ideas from secure multiparty computation can provide strict security guarantees for the implementation of these techniques. We present evaluations against current practices for our proposed solutions and rigorous security proofs for our protocols as we play our role in the evolution of the American healthcare system.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

I am eternally grateful to the following people, each of whom has been instrumental in making this thesis a reality.

First, I would like to express gratitude to my advisor, Dr. Martine De Cock. Not only was her knowledge and experience instrumental in the success of each publication, but her advice and support throughout my time in the Master's program has been invaluable. I have been extremely fortunate to have had Martine in my corner for the past two years.

I would also like to acknowledge the guidance and encouragement of Dr. Anderson C.A. Nascimento, without whom I would have never entered the world of security. Anderson's expertise and inquisitive nature have been a driving force behind my success as a researcher.

My appreciation also extends to Dr. Ankur Teredesai and Dr. Shanu Sushmita without whom I would not have become an integrated member of the Healthcare Analytics research group behind many of the results presented here. A special thanks also goes to those in the Center for Data Science who have provided willing collaboration when needed, especially James Marquardt and Aftab Hassan.

I would also like to express my deepest appreciation for my parents, Allison Smith and John Newman, for their support in all endeavors I have ever dreamed to embark on, my fiancé William Truex for his unconditional love and willingness to make an extra pot of coffee on late nights, and my most loyal canine companion, Lincoln, for keeping me company on the couch for each of those late nights.

This thesis would not have been possible without each individual above and the support of both the Institute of Technology and the Center for Data Science at the University of Washington Tacoma.

DEDICATION

To my loving fiancé, **William**,

for his support, encouragement, and critical eye as a proofreader.

Chapter 1

INTRODUCTION

According to the World Health Organization, healthcare costs in the United States rank highest in the world, yet, amongst comparable nations, the United States ranks poorly in terms of quality of care [8]. To target and encourage improvement in these areas, healthcare reform policies are currently underway which promote initiatives for managing the overall health of a population while keeping costs reasonable [1]. Predicting whether a patient is at risk of being readmitted to the hospital within 30 days after discharge and healthcare cost prediction are both extremely important problems as the United States aims for this improvement in accountability of care. Within the current system, patients with chronic conditions are repeatedly getting admitted to a hospital for treatment and care and being discharged when their condition appears to stabilize, only to get readmitted again, often within just a few days. This kind of readmission problem is severe within the U.S. with 20% of Medicare patients currently being readmitted within 30 days of discharge and 75% of these readmissions being considered avoidable [4]. In addition to highlighting shortcomings in care quality, hospital readmissions also place a huge financial burden on the health system with avoidable readmissions accounting for around $17 billion a year [4]. Using machine learning techniques to solve these problems gives healthcare organizations the ability to prioritize a care plan along both readmission risk and future cost and can enable more effective allocation of limited human and budgetary resources to improve overall population management.

These medical and data mining problems involve large, heterogeneous patient populations and challenging datasets, but are important problems to address as we seek to improve the overall healthcare system within the United States. Seeking solutions to these problems poses important challenges in leveraging existing large and varied clinical and claims datasets to

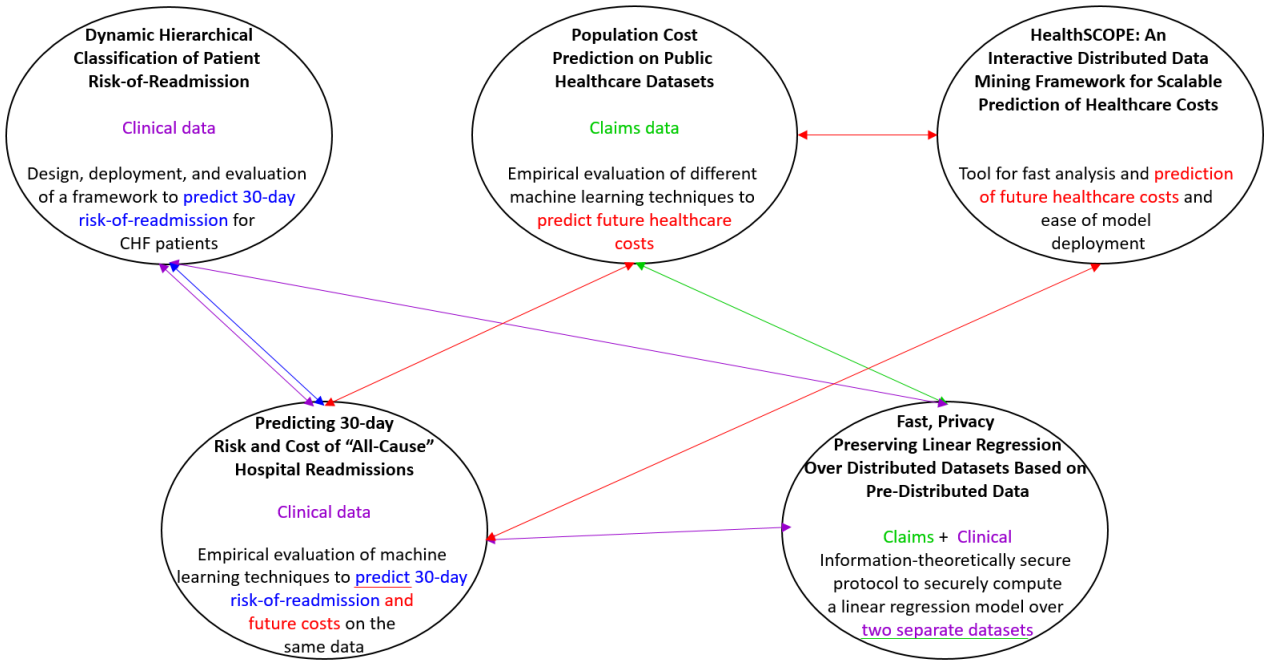

Figure 1.1: Diagram of Proposed Solutions

estimate future healthcare costs and guide care-management with the overall goal of reducing costs while improving overall population health. Additionally, these datasets, used in creating predictive models within the heathcare domain, are often voluminous, diverse, and vary significantly over time. As the amount and complexity of data increases, so does the challenge to store, retrieve and manipulate. Handling such diverse, voluminous, and dynamic data remains a challenge within healthcare analytics and for many healthcare organizations. As storage and computing becomes increasingly distributed and heterogeneous, data movement and transformations become non-trivial, making conventional ML algorithms difficult to apply. Data which gives the full picture of a patient, a picture necessary to power accurate machine learning models, is often held by multiple parties – healthcare providers, hospitals, and medical insurance companies – who often do not want to or legally cannot share their data, creating a dire need for privacy preserving ML techniques. Unfortunately, techniques for constructing popular machine learning models, such as the linear regression model, require

that all training attributes and tuples be accessible to the learning algorithm, which is often not the case in the healthcare setting. In this thesis we focus on tackling the task of providing meaningful and useful tools for providing accountable care from a multitude of angles to address each of these problems. The relationship between each of our contributions to improving the healthcare system is outlined in Figure 1.1.

In Chapter 2 we begin tackling the first task in providing accountable care: improving care quality. We propose a dynamic hierarchical classification (DHC) framework to predict 30-day hospital readmissions for heart failure patients and a deployment of our DHC framework. Then Chapters 3 and 4 address the other prong in accountable care: cost management. We investigate predicting future healthcare costs in Chapter 3 and in Chapter 4 create the tool HealthSCOPE for both patients and healthcare organizations to take advantage of our predictive models. We also explore in Chapter 5 the value of different machine learning algorithms for the prediction of both readmission and cost across the same feature set within the clinical data collected by the MultiCare Health System. Finally, in Chapter 6, we propose a protocol to securely learn a linear regression model across distributed datasets, a solution which would allow for more accurate model building in a security-conscious domain such as healthcare, opening the door for both the DHC framework and the HealthSCOPE tool to be powered by more accurate predictive models which leverage data across multiple organizations.

In our readmission work in Chapter 2 we propose the DHC framework, a framework which predicts a patient's risk of readmission, a key care quality metric. In our work we focus on the Congestive Heart Failure (CHF) cohort but the DHC framework remains extensible to predict readmissions for any population. We aim to provide healthcare providers with a reliable tool which can identify patients at high risk for readmission who may need intervention. We integrate our framework into a local hospital system and outline the challenges behind this implementation. In addition to this highly-tuned readmission classification framework, we also explore predicting readmission across all patients, rather than the CHF cohort in isolation, using features more broadly available. This work is detailed in Chapter 5 where we

aim to use the same feature set to predict both readmission and the cost of that readmission, allowing providers insight into battling both readmissions and hospital costs. We provide a comprehensive analysis of the success of different machine learning techniques in tackling this problem. Our work has provided both solutions and important insights into the problem of hospital readmissions for improving quality of care in the health system.

In addition to readmission, this thesis also addresses the problem of rising costs in the United States healthcare system. We provide an analysis of the value of machine learning techniques when predicting a patient's costs for a coming time frame in Chapter 3. We show that state of the art machine learning approaches outperform the statistical methods which were previously used in the cost prediction domain. We then take our work to the next level in Chapter 4 by exposing the value of these machine learning models in our web based application, HealthSCOPE, which targets both insurers and individual patients. In this work we provide a tool which enables insurers to investigate the causes of high costs within their population as the medical domain continues to battle the problem of rising healthcare costs. The cost prediction problem is then pivoted from the claims point of view to the clinical point of view as we aimed to predict the cost of a patient's next admission. As previously mentioned, Chapter 5 uses the same clinical data from the MultiCare Health System as was used to predict readmission to analyse the performance of different regression techniques in predicting cost. Through our extensive work we have allowed providers and insurers to investigate the causes of high medical costs and revealed the impact machine learning techniques and approaches can have in this domain.

Throughout our work in cost and readmission prediction we have constantly seen the impact of data on the accuracy of our models and experienced first hand the many barriers to accessing the necessary data for these models in the privacy-conscious healthcare domain. While we produced results in cost prediction over both claims and clinical data, there was no way to aggregate information across the two to leverage both information types and further increase model accuracy. To address this issue, we produced solutions in the area of privacy-preserving machine learning. In Chapter 6, we propose a fast, privacy-preserving

linear regression protocol which enables two parties, such as a hospital and an insurance provider, to learn a linear regression model which leverages information across both parties' datasets without either dataset being leaked. We provide proofs of correctness and security for our protocols and experimental results across datasets of varying sizes. Our work is a marked improvement over previous results in the area and is efficient enough to be usable for real healthcare institutions. Through all of this combined work we have addressed three of the biggest challenges facing the healthcare domain today: quality, cost, and privacy. This thesis will detail the contributions made in these areas as we join the initiative to improve and revolutionize the healthcare domain.

The main contributions we make in this thesis are:

- design, deployment, and evaluation of a dynamic hierarchical classification framework to predict 30-day risk of readmission for CHF patients

- empirical evaluation of different machine learning techniques to predict future healthcare costs across multiple datasets and future scenarios

- a tool, HealthSCOPE, which allows for fast analysis and estimation of cost as well as ease of predictive model deployment

- empirical evaluation of different machine learning techniques to simultaneously predict cost as well as risk of readmission using the same data for all patients, including those outside well studied cohorts

- an information-theoretically secure protocol to securely compute a linear regression model over two separate datasets which is both proven to be secure as well as shown to run in a reasonable amount of time

The results of this thesis have been published in a variety of venues related to data mining and security:

- Chapter 2 has been published in the Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2015).

- Chapter 3 has been published in the Proceedings of the 5th International Conference on Digital Health (DH2015)

- Chapter 4 has been published in the Proceedings of the 2014 IEEE International Conference on Data Mining Workshop (ICDMW2014)

- Chapter 5 has been published in the Proceedings for the 30th AAAI Conference on Artificial Intelligence Expanding the Boundaries of Health Informatics Using AI Workshop (AAAI2016)

- Chapter 6 has been published in the Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security (AISec2015)

Chapter 2

DYNAMIC HIERARCHICAL CLASSIFICATION FOR PATIENT RISK-OF-READMISSION

Dynamic Hierarchical Classification for Patient Risk-of-Readmission

Senjuti Basu Roy[†], Ankur Teredesai[†], Kiyana Zolfaghar[†],
Rui Liu[†], David Hazel[†⋆], Stacey Newman[†], Albert Marinez [⋄].
[†] Center for Data Science, UW Tacoma, [⋄] MultiCare Health System, [⋆] Kensci Inc.,
{senjutib,ankurt, kiyana, ruiliu, dhazel, newmsc8}@uw.edu,
albert.marinez@multicare.org, david@kensci.com

ABSTRACT

Congestive Heart Failure (CHF) is a serious chronic condition often leading to 50% mortality within 5 years. Improper treatment and post-discharge care of CHF patients leads to repeat frequent hospitalizations (i.e., readmissions). Accurately predicting patient's *risk-of-readmission* enables care-providers to plan resources, perform factor analysis, and improve patient quality of life. In this paper, we describe a supervised learning framework, *Dynamic Hierarchical Classification (DHC)* for patient's risk-of-readmission prediction. Learning the hierarchy of classifiers is often the most challenging component of such classification schemes. The novelty of our approach is to *algorithmically* generate various layers and combine them to predict overall 30-day risk-of-readmission. While the components of DHC are generic, in this work, we focus on congestive heart failure (CHF), a pressing chronic condition. Since healthcare data is diverse and rich and each source and feature-subset provides different insights into a complex problem, our DHC based prediction approach intelligently leverages each source and feature-subset to optimize different objectives (such as, Recall or AUC) for CHF risk-of-readmission. DHC's algorithmic layering capability is trained and tested over two real world datasets and is currently integrated into the clinical decision support tools at MultiCare Health System (MHS), a major provider of healthcare services in the northwestern US. It is integrated into a QlikView App (with EMR integration planned for Q2) and currently scores patients everyday, helping to mitigate readmissions and improve quality of care, leading to healthier outcomes and cost savings.

1. INTRODUCTION

Hospital readmissions have come to the forefront of healthcare research and discussions in recent years for their recognized universal negative impacts on healthcare systems' budgets and patient loads throughout the world. Within the United States, Centers for Medicare & Medicaid Services (CMS) recently began using readmission rates as a publicly reported quality metric to measure hospital care standards and reimbursements in the fee-for-service model[1]. The severity of the readmissions problem can even be measured in economic terms: the estimated cost of unplanned readmissions is roughly $17.9 billion per year [8]. A significant percentage of these costs are attributable to patients who are often the sickest and most vulnerable: old, critically ill, and suffering from multiple chronic disease conditions. Paradoxically, more than 27% of such readmissions are avoidable [16]. Hence, predicting risk-of-readmission can guide implementation of appropriate interventions to prevent such avoidable readmissions. To that end, this paper investigates the issues faced and challenges addressed in developing and deploying a *framework to predict the risk-of-readmission (RoR)*.

While the deployed framework is extensible by design and includes other chronic conditions, in this paper the chronic condition we focus on is Congestive Heart Failure (CHF) since CHF is one of the leading causes of hospitalization. Studies also indicate that a large percentage of CHF admissions are actually readmissions within a short window of time. The 2005 data for Medicare beneficiaries estimated that 12.5% of Medicare patients admitted due to CHF were followed by readmissions within 15 days, accounting for about $590 million in healthcare costs [12]. In practice, a window of 30-days after discharge for CHF is considered *clinically meaningful* for hospitals and medical communities to take action to reduce readmissions [11] and forms the basis of most readmission risk prediction models.

Current research treats the problem as a binary classification task, where the objective is to identify patients with CHF who are likely to be readmitted within 30 days of discharge as an output of a single binary classifier [21, 19, 18]. A patient readmitted within 30 days = 1 and a patient not readmitted within 30 days = 0. There are several shortcomings to this approach which we address with the DHC framework: the distribution of risk of readmission is highly skewed with a few patients readmitting repeatedly and many patients readmitting once or twice over a period of a year. Traditional classifiers suffer from majority bias and tend to assign patients to the no 30-day-readmission majority class. Moreover, patient characteristics change over time. New diseases and conditions set in; age and vitals continuously change. Thus, including all patients discharged with CHF in the training set to build the classification model introduces noise since patients that were readmitted after a long gap can have characteristics that are very different from pa-

KDD'15, August 10-13, 2015, Sydney, NSW, Australia.

DOI: http://dx.doi.org/10.1145/2783258.2788585.

[1] http://www.cms.gov

tients that were readmitted within a short time (30 days) for the same condition. Hence the team of data scientists and the underlying framework has to handle numerous missing values, discretize attributes, collate comorbidities, extract suitable features and employ the right learning algorithm given the constraints.

This research presents a *Dynamic Hierarchical Classification (DHC)* framework to predict RoR for CHF patients. The primary novelty of the solution is to undertake the prediction problem in *several stages or layers*, leading to the formation of a hierarchy of classification models as the overall solution. Each layer aims at predicting the RoR within *certain days* (cut-points). For example, it may be better to predict whether a CHF patient is likely to (ever) readmit or not before calculating her readmission risk within 30 days of discharge. Each such logical consideration constitutes *a classification layer*. Within a given layer, the problem is of a binary classification design layer by layer[2]. Consider a 70 year old female patient suffering from primary diagnosis of congestive heart failure and various associated conditions such as diabetes and renal failure. If at layer one, DHC predicts her to be likely to be readmitted (ever), then the second layer may predict likely to be readmitted within-60-days of discharge (or not). Subsequently, only when she qualifies to be readmitted with a high confidence at both these layers, would the third layer predict her likelihood to be readmitted within 30 days (or not). This may sound simple conceptually, but the challenge of *how many layers are sufficient and how to design the intermediate layers* remain an open problem. Furthermore, the effectiveness of different features may be different in various layers. Then, the question is, how to select the features? We address this by performing layer specific feature selection. The last challenge is how to perform effective binary classification in each layer. We propose threshold tuning of the classification models for that.

In spite of algorithmic automation as proposed, it is not easy to design these layers and always get it right. Clinical teams may provide initial guidance but there is no clear clinical evidence for how many layers (for decisioning) are appropriate for a given patient population or how to define the time-window for readmission cut-points. Naturally, many of these design decisions are also dataset specific, requiring us to discover these layers for a given patient cohort. In this work, we propose an innovative approach: we design multiple algorithms to discover the actual cut-points assuming that the number of layers is specified as an input. In one of our algorithms, we primarily analyze the characteristics of the underlying patient population and propose greedy algorithms to design the cut-points such that the two consecutive layers generated by our algorithm exhibit the highest difference in the characteristics of the patient population. We also non-trivially adapt one of the frequency based popular discretization algorithms, Chi-Merge [7] (that is primarily used to discretize continuous attributes based on class distribution), to generate these cut-points. After that, to maximize a particular metric (such as AUC, precision, recall, accuracy, etc.), we study how to make use of the training data and the trained classification model in a given layer. We propose novel solutions to the above mentioned problems in this work.

[2]This could be formalized by finding the maximum duration X between two admissions in the dataset; i.e.the task is to predict whether the patient would be readmitted with X days or not.

We present comprehensive experimental results using two real world datasets - we consider 4 years of the State Inpatient Dataset (SID) of Washington State as our large-scale dataset and then we use the real patient dataset provided by MultiCare Health System, a major health system in the northwestern US as the small scale dataset. We empirically evaluate the effectiveness of our proposed solutions with statistical significance analysis using a variety of quality metrics and perform comparative analyses with our hierarchical classifications models against several baseline algorithms. Our results demonstrate that the proposed framework is superior to the baseline algorithms for all quality metrics with statistical significance.

To summarize, we make the following contributions:

1. We initiate the study of designing hierarchical classification models to predict 30-day RoR for CHF patients (Section 2).
2. We propose a *deployed solution for Dynamic Hierarchical Classification (DHC)* framework, empowered with multiple novel solutions to effectively design the prediction hierarchies (Section 3 and and 4).
3. We perform comprehensive experiments using multiple real world patient data sets, demonstrating that the proposed hierarchical model enhances the quality of prediction (Section 5).

2. PRELIMINARIES

It is typical in clinical settings for data to be spread across various relational schemas and flat files; a series of (anonymized) patient record data for various windows of time and modalities of measurements are typically extracted from EMRs and secondary hospital sources to form a data warehouse with appropriate dimensions and measures. Such a schema contains granularities of patient measures related to admission and discharge, other diseases, demographic factors, comorbidities, and post-discharge and follow-up-plans, which are all integrated and preprocessed. Each encounter is uniquely identified by an admission ID. Multiple admissions (i.e., readmissions) of the same patient are identified by the same patient ID and different encounter ID. For a given patient and a given hospital admission, we can calculate the number of days between the current admission and her previous discharge and check if it is less than 30.

Problem definition 1. Given a patient record, the problem of predicting the RoR within γ days of discharge is a hierarchical classification problem. The intuition here is to first predict the readmission window for higher γ values before predicting readmission for $\gamma = 30$ days. Each of these *time intervals* are cut-points, demarcating a classification layer in dynamic hierarchical classification (DHC). Since the 30 day interval is clinically meaningful and tied to reimbursements, that interval is the last layer of risk scoring and outcomes. Each layer can constitute a specific readmission window (for example the very first layer predicts whether a patient would be readmitted at all or not for γ >365 days in a year). Each layer design is to formulate a binary classification model using historical patient data relevant to that interval to predict the RoR within that readmission interval. How many such layers are to be designed, and how to decide the readmission intervals algorithmically, is the key contribution of this work and described in depth in Section 4.

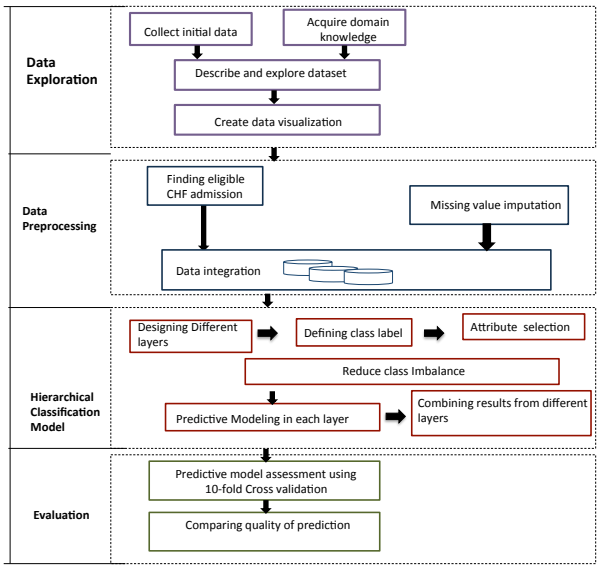

Figure 1: Overall architecture for RoR prediction process includes various stages as seen above including: exploration, preprocessing, modeling, and evaluation.

Data exploration, data pre-processing, hierarchical classification model, and model evaluation are critical for a successful data mining framework, especially at healthcare domain [21, 19, 18]. Figure 1 provides an overview of these major steps. For the DHC framework design we worked closely with cardiologists, nurse practictioners, and other members of the care team to identify critical factors influencing early recurrent readmission. The robustness of these factors and their availability at various points in the care process was an important DHC design consideration.

3. DEPLOYING THE DHC FRAMEWORK

Prior research efforts have focused on the accuracy of modeling to predict the likelihood of patients with chronic conditions to readmit within 30 days[15]. Commercial efforts have been limited and mostly restricted to web based mortality tools or risk calculators where data is manually entered to compute the risk. To the best of our knowledge, ours is the first cloud based scalable EMR integrated effort reported in literature that uses data mining fundamentals to continuously monitor risk of patients and issue risk scores to care providers. We have enabled the platform to deliver the per-patient readmission risk score along with actionable insights, not solely as a business intelligence tool sourced from a data warehouse (see 3.1), but on real-time clinical data, delivered within the clinical workflow, at the point of care.

The DHC framework consists of three main components: a) External Layer, b) Communications Layer, c) Analytics Layer. Within these layers there are multiple subcomponents and processes which are tightly integrated in order to accurately and efficiently score patients for 30 day risk of readmission and deliver these risk profiles to the patients electronic medical record (EMR).

3.1 QlikView Application

The initial deployment scenario envisioned was a single layer binary classifier exposed through a QlikView Readmission App[3]. This would provide system-wide readmission visualization and reporting with facility level, cohort level, and patient level drill down capability. Additionally, the QlikView Readmission App was used to identify and flag patients (within targeted risk categories) for further investigation and focus. It enables exploration of a detailed patient risk profile (30 day risk of readmission score targeting clinical attributes, psychosocial factors, a composite score, and top contributing factors). With the development of the DHC framework, the QlikView risk scores are now computed by the hierarchical classification with algorithmic multi-layer design. A spapshot of the application is presented in Figure 2.

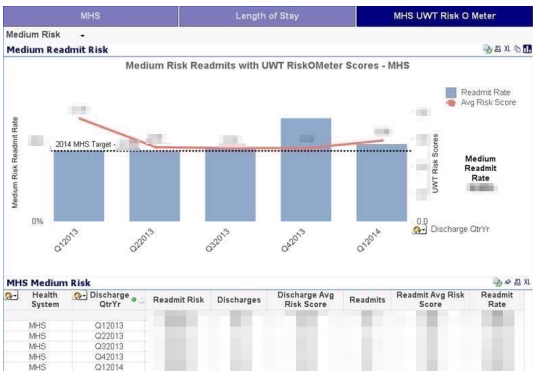

Figure 2: Snapshot of the QlikView Application

Every night it scores discharged patients and creates a ranked list of readmission risk for further review.

3.2 EMR Integration

Business intelligence tools 3.1 are useful for population level reporting and risk management but are not oriented towards the clinical workflow and have limited adoption within that context. To fully leverage our developed framework for CHF readmissions, cardiovascular service line leadership recognized the need to deploy DHC directly into their EMR (Epic).

After extensive user research, the team identified two target Epic integration points: 1) the Heart Failure Dashboard, used by cardiology to see the overview of the patients and help drive interventions during the encounter, 2) Care Management Doc Flowsheet, used to support discharge planning (which begins when the patient is admitted) and bring focus to psychosocial factors and followup care.

The EMR integration roadmap consists of: 1) publishing of a single combined score (clinical + psychosocial) as a "result" [4] in Epic, 2) subcomponents of that result to include discrete clinical and psychosocial scores.

3.3 Patient Data Flows

DHC is available as an on premise deployment or as a series of services hosted on Microsoft Azure. The overall architecture of the system and flow of messages are illustrated in Figure 3.

[3]http://www.qlik.com/us/explore/products/qlikview

[4]In the same way that lab results can be tracked and recorded over time, the insertion of the score as a result, with primary value and multiple sub-components, enables comprehensive trend analysis, reporting capability and intervention exploration.

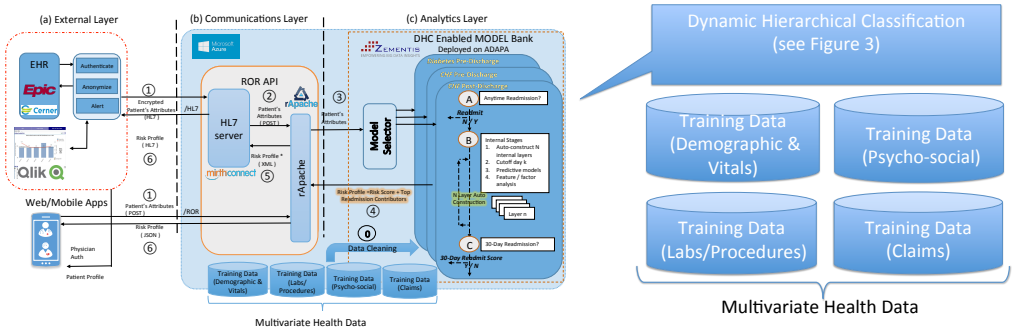

Figure 3: Illustration of the DHC Framework: (a) External Layer, (b) Communications Layer, (c) Analytics Layer. The focus of this paper is the Dynamic Hierarchical Classification in (c) Analytics Layer.

We now return to our example of a 70 year old female patient. She has been admitted to the inpatient setting and within 24 hours her lab results become available. As her medical record within Epic is updated, an HL7 message [5] containing the relevant attributes is generated. This message is then sent from the External Layer (a) to the Communications Layer (b). For cloud deployments, where the data does not stay fully resident within the health system, the tuple of patient data is passed through an anonymization service, which sends only the attributes (+GUID) required for scoring to the API [15].

The HL7 message is then ingested by Mirth Connect service [6] which handles message queuing and translation from HL7 into JSON, used internally by the DHC Enabled Model Bank. For native JSON requests (such as from a mobile device), we would bypass Mirth Connect [15].

After ensuring proper formatting and consumption of the message, it is passed to the Analytics Layer for scoring. We use the Zementis ADAPA Scoring Engine [7] and the DHC Enabled Model Bank. The use of ADAPA brings multiple benefits, among these the ability to scale up to handle a very large volume of messages, as well as the ability to import multiple models (beyond those we have developed for CHF), allowing us to score patients against a number of chronic disease conditions. Additionally, the DHC supports auto selection of cut-points, where the model is trained against MultiVariate Health Data sources outlined in Figure 3 (the significance of this is outlined in Section 4). When a patient tuple is passed to the APIs provided by the ADAPA scoring engine, it is scored against the DHC Model which has been trained for a particular cohort, such as CHF Post-Discharge.

A risk profile, consisting of a risk score as described next in section 4.3 and top correlated factors (identified using Chi-Square test) that contribute to readmission risk, is then generated and returned back to the Communications Layer. In the case of Epic deployment, the JSON response is converted back into HL7 for consumption. This is then returned to the anonymization service, where the risk profile is recoupled with the patient identifier, attached to the patient visit, and published as a result within Epic.

[5]HL7: Messaging format(standards, guidelines and methodologies) by which various health care systems communicate.
[6]http://www.mirthcorp.com/products/mirth-connect
[7]http://zementis.com/products/adapa/

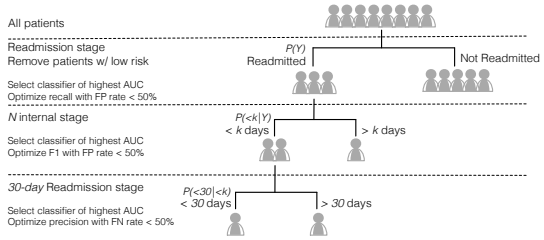

Figure 4: Tree-based hierarchical class structure for predicting 30-day readmission using 1 intermediate layer.

4. DYNAMIC HIERARCHICAL CLASSIFICATION (DHC)

We propose *Dynamic Hierarchical Classification (DHC)* to predict the risk of readmission within 30 days, transforming the problem into a tree-structured class hierarchy. Our approach produces a hierarchical set of classifiers in top-down fashion. The hierarchical setting provides the opportunity to make more specific and accurate predictions and improve knowledge about the problem using factor analysis (Chi-squared test). The framework also allows flexible choices of classifiers at different layers. As the distribution of data changes at each layer, DHC allows us to use the best classifier to optimize different learning objectives for each layer.

For our problem (i.e., predicting 30-day RoR), the hierarchical classification task could be restated as follows: first predict if a patient would ever be readmitted or not. For that task, a binary prediction model is designed using appropriate training data. Then, in the *intermediate layers, a set of classifiers* needs to be designed with each constituting a particular readmission window. Each of these is a subtask towards predicting RoR with a specific time window (cut-point). We design binary classification models for each of these intermediate layers. For simplicity, we assume that the number of intermediate layers is predefined and given [8]. The final layer of the classification is to predict whether a patient would be readmitted within 30 days or not. Similar to all previous layers, this sub-task is also treated as a binary classification problem and the appropriate training dataset is used for that purpose.

[8]However, in our experimental analyses, we empirically decide the number of intermediate layers.

For the purposes of illustration, Figure 4 describes the overall process of predicting 30-day RoR using 1 intermediate layer. The framework involves three stages: *Readmission Stage*, *Internal Stage*, and *30-day Readmission Stage*. Even though the number of layers in the internal stage are given, we still have to identify *the appropriate readmission windows, i.e., cut points* for those layers. Our proposed solutions to design cut-points are described in Section 4.2.

Algorithm 1: DHC Algorithm

Require: Training dataset D, number of internal layers N, cutoff day $k > 30$, Max is the maximum readmission time based on the dataset D

1: `[Stage 1]` Design Max-day readmission prediction model using D
2: $m = 31$, $n = Max$
3: **for** $i = 1$ to $N - 1$ **do**
4: Extract $D' \subset D$, where D' contains only patients who are readmitted within (m, n)-days in D
5: $k \leftarrow$ CutpointFinder(D', m, n)
6: `[Stage 2]` Design (m, k)-day readmission prediction model using D'
7: $D \leftarrow D'$
8: $m = k + 1$
9: Design (m, Max)-day readmission prediction model using D', where D' contains only patients who were readmitted within (m, Max) days in D
10: Extract $D'' \subset D$, where D'' contains only patients who were readmitted within $(0, 30)$ days in D
11: `[Stage 3]` Design $(0, 30)$-day readmission prediction model using D'

Algorithm 1 formalizes the DHC framework. Similar to the multistage screening in medical practice, we impose a maximum error bound while maximizing the recall and precision at different stages. The error bound (i.e. false positive rate) is determined by domain experts. Table 1 provides a tabular view of *Readmission Stage*, *Internal Stage*, and *30-day Readmission Stage*.

4.1 Readmission Stage

The first *Readmission* stage predicts whether a patient will be readmitted at all after being discharged from the hospital. The learning objective is to maximize AUC and recall [7] in order to capture all patients with readmission risk. The maximum false positive rate is set to be 50%. Since this is the root (very top level) of the hierarchical classification model, the goal is to filter out patients who are unlikely to be readmitted.

4.2 Internal Stage

The objective is to maximize AUC and recall while controlling the false positive rate to be lower than 50%. We build N internal layers and, for each layer, derive a cutoff day k. Even when N is a given integer number, we still have to determine N different readmission windows (cut-points). Intuitively, what we intend to do is to choose those cut-points, such that, if one can *create N non-overlapping partitions of the entire patient population based on those cut-points, patients that fall across the stratum are as divergent as possible*. We propose a heuristic algorithm towards that end which iteratively creates N different cut-points.

Algorithm 2 describes the *hill climbing* search heuristic to identify a single cutoff day k that constitutes a layer. Without loss of generality, to find a cut-off day k between a time window (m, n) (note that if $N = 1$, then $m = 31$, $n = Max$, where, for a given dataset D, the highest readmission day is denoted to be Max), we call a pseudo-random number generator between (m, n) to get a k, such that $m < k < n$. We select two candidate cut-points: $k' = k + 5$ and $k'' = k - 5$. After that, we compute two divergences: one based on k' and another based on k''. k then gets updated based on whichever cut-point gives rise to higher divergence. We continue the search until no further improvement is possible. Once a single cut-off point k is found, we run Algorithm 2 twice to find two other cut-off points, one between 31 and k and the other between $k + 1$ and Max. This process is repeated until all N internal layers are identified. Next, we describe two divergence calculation methods used by Algorithm 2.

4.2.1 Entropy-based Divergence Methods

Recall Algorithm 2 and note that to select a single cut-point between (m, n) window, the hill climbing algorithm needs to compute the divergence of patient population that constitutes two readmisison windows between (m, k) and $(k + 1, n)$ based on any k, $m < k < n$. In order to use Entropy-based divergence methods, one has to represent each of these patient populations as a probability distribution. To do that, we obtain the *center of the individual patient group*, where the center is the *average* for the numeric attributes and *mode* for categorical attribute, as described in Algorithm 3 (DIVCAL).

In our implementation, we consider two popular divergence methods: Kullback-Leibler Divergence (KL) and Jensen-Shannon Divergence (JSD) [3]. The main difference between the two methods is that unlike KL, JSD is symmetric and always defined. The divergence between two centers p and q of m dimensions, for Kullback-Leibler(KL) and Jensen-Shannon(JSD) methods are calculated as follows [3]:

$$KL(p, q) = \Sigma_{\forall i=1}^{m} ln(\tfrac{p(i)}{q(i)}) \times p(i)$$

$$JSD(p, q) = \frac{1}{2}\left(\Sigma_{\forall i=1}^{m} ln(\tfrac{2p(i)}{p(i)+q(i)}) \times p(i) + \Sigma_{\forall i=1}^{m} ln(\tfrac{2q(i)}{p(i)+q(i)}) \times q(i)\right)$$

Algorithm 2: Subroutine CutpointFinder(D, m, n)

Require: Dataset D, lower boundary of readmission window m, upper boundary of readmission window n

1: $k \leftarrow$ RANDOM_NUMBER(m, n)
2: **repeat**
3: $k' \leftarrow k + 5$
4: $k'' \leftarrow k - 5$
5: currentDIV $= DIVCAL(k)$
6: **for all** $i \in \{k', k''\}$ **do**
7: **if** $DIVCAL(i) >$ currentDIV **then**
8: $k = i$
9: currentDIV $= DIVCAL(i)$
10: **until** $DIVCAL(k) > DIVCAL(k')$ and $DIVCAL(k) > DIVCAL(k'')$
11: **return** k

4.2.2 Chi-Merge Based Solutions

In addition to Entropy-based methods, we intelligently adapt the popular frequency based Chi-Merge algorithm [10] to design the internal layers.

	Readmission Stage	Internal Stage	30-day Stage
Objective	AUC & Recall	AUC	AUC & Precision
Error Bound	FP Rate $< 50\%$	FP Rate $< 50\%$	NA
# of Layers	1	N	1
Probability	$P(Y)$	$P(< k\|Y)$	$P(< 30\| < k)$

Table 1: Comparison of three stages

Algorithm 3: Subroutine DIVCAL(k)

Require: Dataset D, cutoff day k, center of data V
1: $D' \leftarrow$ subset of D where readmission day between m, k
2: $D'' \leftarrow$ subset of D where readmission day between $k+1, n$
3: $p \leftarrow$ center of D'
4: $q' \leftarrow$ center of D''
5: $d \leftarrow KL(p,q) Or JSD(p,q)$
6: **return** d

Chi-Merge is traditionally used to discretize continuous attributes based on the class distribution frequency. On the other hand, our objective is to create a set of N intermediate layers (and cut-points thereof) for the variable readmission, where this variable itself is also the class label. This precludes direct adaptation of Chi-Merge in our settings. In fact, to be able to apply Chi-Merge, we need to extend our settings in the following way that we illustrate using a simple example next. Imagine that we have a population of 100 patients in total. Out of these 100 patients, 20 got readmitted within 30 days and 10 never got readmitted. The remaining 70 patients create the positive instances who got readmitted within 30 and Max days. Similarly, Chi-Merge also needs negative instances. If a set of X' patients are readmitted on a day d, then we set the negative instances using the remaining $70 - X'$ patients. After this modification, we apply Chi-Merge to decide the N intermediate layers.

4.3 30-day Readmission Stage

The final stage predicts the readmission ≤ 30 days. The sample size has been reduced as we move down toward this final leaf node. We use local information from this subset to train classifiers and assess the likelihood of a given patient being readmitted to the hospital within 30 days – i.e. $P(Readmit \leq 30 | Readmit \leq k)$. The objective of this stage is to maximize the AUC and precision so that the medical resources can be efficiently invested on the right patients.

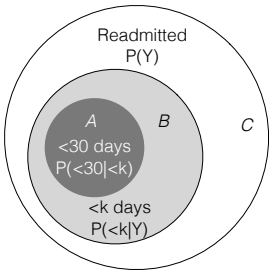

Figure 5: Relationships of predicted probability of three stages considering 1 intermediate layer

Figure 5 shows the relationship of predicted probability of three stages, where circle A is the final stage, circle B is the internal stage (for simplicity we consider 1 internal stage in this example), and circle C is the readmission stage. Intuitively, the predicted probability of 30-day readmission should be the highest in A, lower in B, and the lowest in C. Given the definition of the problem for predicting 30-day readmission, the goal is to estimate $P(\leq 30)$, as below.

$$P(\leq 30 | \leq k) = \frac{P(\leq 30 \wedge \leq k)}{P(\leq k)}$$

Since ≤ 30 is a subset of $\leq k$, we obtain:

$$P(\leq 30 \wedge \leq k) = P(\leq 30)$$

Therefore,

$$P(\leq 30) = P(\leq 30 | \leq k) P(\leq k)$$

Similarity, we can obtain $P(k)$ as follows:

$$P(\leq k | Y) = \frac{P(\leq k \wedge Y)}{P(Y)}$$

Since $\leq k$ is a subset of $Readmission = Y$, we obtain:

$$P(\leq k \wedge Y) = P(\leq k)$$

Thus,

$$P(\leq k) = P(Y) P(\leq k | Y)$$

Therefore,

$$P(\leq 30) = P(\leq 30 | \leq k) P(\leq k) = P(\leq 30 | \leq k) P(\leq k | Y) P(Y)$$

Given the equations above, we can compute $P(\leq 30)$ for all instances in A in Figure 5. We then combine the results using normalization techniques to ensure the correct sequence of instances: $A \prec B \prec C$, if the list is sorted in descending order by the predicted probability. In Figure 5, we identify three mutually exclusive areas – the dark grey area in A, the light grey area in B, and the white area in C. For all instances in A (the dark grey area), we normalize the predicted probability in to the range (0.7,1]. For all instances in the light grey area in B, we normalize the predicted probability to the range (0.3,0.7]. Finally, we fit the predicted probability from the white area in C into [0,0.3]. We compute our evaluation metrics (e.g. AUC, recall etc.) using the combined normalized probability vector.

5. EXPERIMENTS AND EVALUATION

We evaluate performance of proposed DHC framework on Washington State Inpatient Dataset and the Heart Failure cohort data from MultiCare Health Systems (MHS).

5.1 SID Dataset

The State Inpatient Databases (SID) [9] of Washington State (referred as SID-WA) is available for four years 2009-2012. It is discharge abstract data that includes inpatient

[9] http://www.hcup-us.ahrq.gov/sidoverview.jsp

discharge records from community hospitals in the State of Washington with all-payer, encounter-level information.

Each year comprises four files that are associated with hospital encounters: core file (CORE), charges file (CHGS), diagnosis and procedure groups file (DXPRGRPS), and disease severity measures file (SEVERITY). Total 596 attributes for each encounter with an unique identifier KEY, that can be used to link records across files (but NOT across years). We construct a heart failure cohort with patients whose primary or secondary ICD9-CM diagnosis codes are heart failure. Our cardiology partners identified a total of 91 attributes to be related to CHF readmission from a mix of demographic variables such age, gender, and race; as well as comorbidities related to diagnosis information derived from primary and secondary diagnosis codes for that patient. Many features related to utilization services such as thew emergency room, ICU, electrocardiograms services and so on are also included in our prediction models.

The SID-WA attribute DaysToEvent is used to compute the days between two consecutive hospital admissions for each patient. The days since previous hospital discharge is computed using the days between two hospital admissions minus the length of stay of the first admission. We exclude admission records in which the patient passed away while in the hospital, and in which the patient age is less than 1 year (age=0). The dataset then contains a total of $2,051,105$ readmission instances.

5.2 Dataset from MultiCare Health System

The Cardiovascular datamart is our primary source within the MultiCare Health System data. It has about 8,600 patients diagnosed with Heart Failure, serviced over $14,200$ hospital encounters from 2009 to 2013. We consider only patients who are discharged to home, excluding inter-hospital transfers. Encounters where the patient expired are not included. The post cleaning and processing set we used for these experiments includes 6,348 patients diagnosed with CHF and 14,170 hospital encounters. A total of 49 attributes were determined to be related to CHF admission. These are mainly clinical. The key socio-demographic factors related to readmissions are: gender, race, and marital status. Other important factors pertinent to CHF are ejection fraction(EF) [10], blood pressure, primary and secondary diagnosis indicating comorbidities, and APR-DRG codes [11] for severity of illness and risk of mortality. Information about discharges, such as discharge status, discharge destination, length of stay and follow-up plans, are also found to be correlated to CHF readmissions. In addition, we include 34 cardiovascular and comorbidity attributes.

5.3 DHC Evaluation

Our modeling for SID-WA data and corresponding analysis is conducted on Microsoft Azure over virtual machines with 16 cores and 112GB of RAM. For the modeling environment we use R and Python both. Models are then exported in PMML and deployed using the Zementis ADAPA scoring engine. We evaluate quality using measures such as Area Under the Curve(AUC), Precision, Recall, and Accuracy [7].

Depending on the final consumption of RoR predicted score for a patient population, different evaluation measures are appropriate. AUC measure is typically interesting when the dataset is imbalanced and we want to ascertain comparison of performance with other competing tools for risk stratification. Precision is critical if there is a high cost related to incorrectly predicting patients' risk to belong to the *Readmission* class with accompanying reimbursement penalties associated with the decision. Recall is relevant if using the DHC framework to identify patients who belong to *Readmission* class from within an overall population for post-discharge care management. Besides these, we also compute the confusion matrix for all the experiments (each fold) consisting of true positive count (TP), false positive count (FP), true negative count (TN), and false negative count (FN).

In general, we select the classifier that renders the highest AUC for each layer for reporting in this paper. In actual deployment we may vary this choice depending on the endpoint where the score is being consumed. We also tune the threshold to optimize the selected evaluation metric for each layer in DHC.

Predictive Models: We use five different predictive models in general. The first one is Logistic Regression (LR) [7], a method that models the outcomes (class labels) as a so-called logit function of the predictive variables. The second one is Naive Bayes (NB) [7], a well-known simple statistical classifier that assumes attribute independence. We also use Random Forests and Adaboost [5], as popular ensemble algorithms (we use 500 decision trees in random forests and 50 CART decision trees in Adaboost). The last implemented model is the Support Vector Machines (SVM) [7] using well-established RBF kernel. For each layer of DHC described in Section 4, we run a 10 fold cross-validation procedure on train set to determine the best classifier.

5.3.1 Comparative study on SID dataset

We follow the holdout method to test the models built on SID-WA data. The data is first partitioned into two independent train and test sets. Three years of data 2009-2011 are allocated to the training set which consists of 1,543,131 admissions and the last year of data (2012), consisting of 507,974 records, is allocated to the test set.

The N internal layers are discovered using either entropy-based divergence calculation methods (Kl-divergence, Jensen Difference), or appropriately adapting frequency based discretization method (Chi-merge), as described in Section 4.

Statistical Significance Test: Wherever applicable, we also perform paired t-test [7] to further understand the statistical significance of the obtained results. To be able to run t-test, we partition the test population into 20 folds, then calculate the quality metric for the competing methods in each fold, and finally run the t-test to investigate whether results are significantly different from each other on each metric. Unless otherwise stated, the significance level is set to p-value < 0.05.

Effect of Number of Internal Layers: We carry out experiments to find the best number of internal layers for each cut-point selection method. We vary the number of layers from 3 to 12. The logic for this range was simply to see if we auto-discover less than 30, 30 to 60, and more than 60 as natural layers. Based on the experiments, the best number of internal layers depends on the cut-point selection methods. However, we observe that the increase in number of layers negatively affects the true positive rate. This

[10] EF is the volumetric fraction of blood pumped out of the ventricle with each heartbeat

[11] All Patient Refined Diagnosis Related Groups Definition

can lead to a significant decrease in recall, while the AUC and the accuracy do not change considerably. The reasoning behind this is as follows: with an increase in the number of layers, more patients are pruned in the initial layers (i.e., predicted readmission is "no") and consequently less patients get promoted to the 30-day readmission layer (which has to be the last layer given the problem formulation). Empirical analyses suggests Chi-Merge to propose 1 (i.e., $N = 1$), and Jensen Divergence optimize at 3 layers overall, while KL-Divergence shows the best performance for 5 internal layers. Table 2 demonstrates the readmission interval for the internal stages as well as the classifier chosen for each stage based on 10 fold cross-validation procedures on training data. Classifiers were chosen based on performance as measured by the appropriate metric from Table 1. It can be seen that Logistic regression (LR) is the dominant classifier for most of the cut-points.

Effect of Cut-Points Selection Algorithms: Recall Section 4 and note that given the number of internal stages as an input, we propose different algorithms to automatically select the N cut-points for the internal stage. The aim of these proposed solutions is to choose cutoff days that discriminates two groups of patients with distinct characteristics as much as possible. As described in previous sections, we vary the value of N experimentally and observe its qualitative effect on different internal hierarchy selection algorithms (Entropy Based Divergence vs Frequencey Based Discretization) and observe that these algorithms lead to different readmission windows (cut-points) for the internal stages. Based on our exhaustive empirical analyses, divergence methods have more stable results in comparison with Chi-Merge, especially when we increase the number of internal stages. Table 3 enlists the best result of each cutoff algorithm based on the best number of layers. The results indicates that, for Recall and AUC, KL Divergence significantly outperforms Jensen Divergence as well as Chi-Merge.

Effect of Threshold Tuning: As discussed in section 4, since the distribution of data changes from layer to layer, we run model selection algorithms at each layer in order to select the classifier that renders the highest AUC. Different data distributions and classifiers at each layer can lead to different probability distribution which may cause some inconsistency in final predictions. DHC allows us to apply scaling, as well as threshold tuning at each stage. This not only solves the problem of varied probability distributions but also enables us to optimize the selected evaluation metric for each stage. The default threshold is set to 0.5. In general, we expect that increasing the threshold throughout the hierarchy will lead to eliminating more patients at upper layers as no-readmission cases. Conversely, decreasing the threshold leaves the classification decision to lower layers. Among evaluation metrics, we focus on recall and AUC, as they provide insight into the performance of classifiers on skewed datasets. We tune the thresholds through the layers such that we improve the overall prediction quality, with the aim of increasing the appropriate objective (see Table 4).

Baseline Algorithms: For comparison, we implement five baseline solutions that employ a single classification model on the full dataset without using a hierarchical structure as proposed in DHC. We consider Logistic Regression, Naive Bayes, SVM, Random Forest, and Adaboost as five classifiers to build our baseline models (prior efforts have demonstrated them to work well for risk of readmission) [21]. Comparing the results in Table 5, the Naive Bayes classifier outperforms other models - it has high AUC, and reasonable recall.

Table 5 compares the result of the baseline models and the best DHC result which was obtained by KL-divergence method with 7 layer (see Table 3 and Table 4). As stated above, we also report the statistical significance of quality results (significance level is set to p-value < 0.05). It can be easily observed that our proposed framework achieves significantly better results for all metrics (AUC, recall, precision and accuracy) compared to the baseline with statistical significance. It is interesting to note that the highest improvement is for the recall and AUC, which are considered as the main metrics when dealing with skewed datasets.

	Random Forests	Adaboost	Naive Bayes
Readmit Stage	0.6803	0.6539	0.6000
Internal Stage	0.6014	0.5830	0.5600
30-day Stage	0.6245	0.5900	0.5833

Table 6: MHS data: Classifier Comparison based on AUC for three stages

5.3.2 *Comparative study on MultiCare Health System dataset*

On MHS internal data, the presented results are representative. We primarily focus on describing the utility of the deployed DHC solution designed using KL-Divergence (as that is the best performing algorithm based on the SID dataset) and compare it with the single stage baseline classification algorithms.

Based on Kl-Divergence, we select one layer in internal stage (i.e., $N = 1$), thereby constituting 3 layers overall (readmit, internal, and the 30-day yes no layer). After getting results from Algorithm 2, the readmission interval for the internal stage was determined to be 65 days. Hence, the problem of predicting 30-day readmission risk is formalized using hierarchical classification as follows: *Readmit Stage*: predicting readmission within 1200 days (based on our dataset, $Max = 1200$), *Internal Stage*: predicting readmission within 65 days, and *30-day Stage*: predicting readmission within 30-days.

DHC Internal Predictive Models for MultiCare Health System: The following three classifiers exhibit the best and somewhat comparable classification performance – Random Forests [2], Adaboost [5], and Naive Bayes [7]– for each layer of DHC described in Section 4. Table 6 shows the AUC of the three classifiers in each layer. As shown in Table 6, Random Forests achieves the highest AUC for all three layers. Therefore, we present the results of Random Forests for all three stages for the rest of our experiments.

Table 7 shows the experimental results for the proposed DHC approach and the prediction quality for each stage. Since, Random Forests achieves the best performance, for brevity, we therefore present our results by comparing DHC with a single-stage baseline Random Forests classifier (500 decision trees). DHC outperforms the baseline on true positive count, recall, and AUC. It is noted that the highest improvement takes place in the *Readmit Stage*, as our predictive model achieves higher AUC at that particular stage. This observation is intuitive as it appears that the quality of prediction can be improved by eliminating negative in-

	layers	Cutpoints
Chi-Merge	3	Readmit(Adaboost),280(LR),30(LR)
KL divergence	7	Readmit(Adaboost),355(svm),240(LR),195(LR),150(LR),110(LR),30(LR)
Jensen Divergence	3	Readmit(Adaboost),280(LR),30(LR)

Table 2: SID Data: Selected cutoffs and classifiers for each cutoff selection methods

	layers	TP	FP	TN	FN	Prec	Recl	Acc	AUC
Chi-Merge	3	34750	103275	329928	40021	0.252*	0.465	0.718*	0.699
KL Divergence	7	38818	118427	314776	35953	0.247	0.519*	0.696	0.701*
Jensen Divergence	3	35762	106907	326296	39009	0.251	0.478	0.713	0.698

Table 3: SID data: Best DHC Results for cutoff selection methods.* denotes whether the result is significantly better than the others among the three variations of cutoff selection methods based on paired-t-test.

stances at an early stage. The overall quality improvement becomes less at the internal stage. This is due to the fact that the data in the internal stage is more homogeneous and no distinct patterns among the patients are identified.

We also have observed that for this dataset the difference in KL-divergence across different k (cut-point) values do not vary greatly (most of them approximately 0.057). After further analysis we realize that such low KL divergence and homogeneity arises due to our use of *mean value* to determine the centroid of each group. In the future we explore alternatives to calculate group centroids, potentially leading to larger values of k for the internal stages. Nevertheless, based on Hill-Climbing we selected $k = 65$.

Factor Analysis: We apply the Chi-Squared test [7] to determine the significant factors. The p-value is set to 0.05 in all cases. We observe that the number of significant factors vary at each stage. There are 32 influential factors for *Readmission Stage* and *Internal Stage* and 16 for *30-day Stage*. Table 8 enlists top-10 most influential factors sorted based on increasing P-values, based on the Chi-Squared test. The importance of a particular factor is different at each stage. For example, *Length of Stay* is an important factor for *Readmission Stage* and *Internal Stage*, but not for *30-day Stage*. It implies that, given that a patient is likely to return to the hospital, care managers can concentrate more on risk factors closely associated to 30-day readmission. Such insights are tremendously useful to the domain experts and the physicians for appropriate prognosis.

Readmission	Internal	30-day
Severity Of Illness	Risk-Of-Mortality	Severity Of Illness
Risk-Of-Mortality	Severity Of Illness	Risk-Of-Mortality
Length-Of-Stay	Length-Of-Stay	Acute coronary syndrome
Renal Failure	Secondary ICD9	Discharge Followup Category
Anemia	Cardio-respiratory Failure	Stroke
Ejection-Fraction	Fluid Disorder	Cardio-respiratory Failure
Pneumonia	Acute coronary syndrome	Fluid Disorder
Fluid Disorder	Discharge Followup Category	Chronic atherosclerosis
Dialysis	Ejection Fraction	Malnutrition

Table 8: MHS data: Factor analysis of three different stages of DHC model

6. RELATED WORK

To the best of our knowledge, no hierarchical classification techniques for risk of readmission risk prediction have been yet reported in literature. Very few cloud based deployed solutions exist today in healthcare and none are for scoring patients for their readmission risk on multiple chronic conditions, in real-time, integrated with EMRs.

Readmission Risk Prediction: An early research result for predicting RoR for CHF patients was termed the Yale Model [11]. Logistic regression was used for 30-day all-cause readmission risk for 65+ year old HF patients. More recently [1], administrative claim data was used to build a regression model on 24,163 patients from 307 hospitals on patients 65 years or older. In collaboration with MultiCare Health Systems, our prior efforts have resulted in accurate solutions [15, 21, 19, 18, 20] to predict 30-day RoR. However, none of these solutions generalizes for any time-interval and automated design, which is one of the primary contributions of the proposed research.

Hierarchical Classification: Hierarchical classification is extensively used in various application areas such as in text mining for web page classification applications [4], Information Retrieval [6], and in signal processing [14]. The use of multistage classification for Bayesian combination of classifiers is explored in literature [13], K-Nearest Neighbor [17], and hierarchical SVMs [9]. Thus we turn to a hierarchical or multi-stage classification process for predicting risk of readmission. The design though is non-trivial to optimize overall prediction quality and ensure good generalization, particularly for high risk patients.

7. CONCLUSION

Predictive models for risk of readmission can significantly improve quality of care. With an increase in the number of patients suffering from chronic conditions, demand for actionable, accurate, and cost-effective solutions to be deployed also increases. In this paper, we describe the algorithm design, deployment challenges, architecture of our

	layers	Thld	TP	FP	TN	FN	Prec	Recl	Acc	AUC
Chi-Merge	3	.55,.55,.45	37921	112700	320503	36850	0.252 †	**0.507***	0.706 †	0.699
KL Divergence	7	.4,.55,.55,.4,.4,.45,.4	40044	121483	311720	34727	0.248	**0.536***†	0.692	0.696
Jensen Divergence	3	.55,.55,.45	39116	118093	315110	35655	0.248	**0.523***	0.697	0.7†

Table 4: SID Data: Threshold tuning for result on table 3.† denotes whether the result is significantly better than the others among the three variations of cutoff selection methods(after threshold tuning).* denotes the result that is significantly higher after threshold tuning compared to result before that. The Thld column describes the selected threshold values from the readmission stage to 30-day stage.

	TP	FP	TN	FN	Prec	Recl	Acc	AUC
logistic regression	52139	264117	169086	22632	0.165	0.697	0.435	0.593
Naive Bayes	37247	120142	313061	37524	0.23666	0.498	0.689	0.678
SVM	3190	2575	430628	71581	0.553	0.043	0.854	0.634
AdaBoost	0	75799	0	432175	NAN	0	0.851	0.696
Random Forests	3150	1434	430741	72649	0.687	0.041	0.854	0.724
DHC (Proposed Technique)	40044	121483	311720	34727	**0.248**†	**0.536**†	**0.692**†	**0.696**†

Table 5: SID-WA data: Single Layer Baseline classifier comparisons. Naive Bayes is the best performing baseline solution.

	Total	Thld	TP	FP	TN	FN	Prec	Recl	Acc	AUC
Baseline (Random Forests)	14,170	0.5	376	139	10,995	2690	0.7301	0.1226	0.8007	0.6492
DHC-combined	14,170	0.5	**1,272**	2,084	9,020	1,794	0.3790	**0.4148**	0.7263	**0.6596**
DHC-Readmit	14,170	0.5	5,119	3,025	3,794	2,232	0.6285	0.6963	0.6290	0.6803
DHC-Internal	8,144	0.46	2,749	4,360	743	292	0.3866	0.9039	0.4287	0.6014
DHC-30Day	6,332	0.7	861	1,230	3,220	1,021	0.4117	0.4574	0.6445	0.6245

Table 7: DHC Results on MHS data: Each stage contains different number of instances (Total column) and uses different threshold (Thld) for classification.

cloud-based deployed framework DHC. MultiCare Health System uses the described DHC framework to predict the 30 day post discharge risk of readmission for their heart failure collaborative. We demonstrate that the proposed framework clearly outperforms baseline solutions for congestive heart failure (CHF). DHC automatically discovers and defines the layers by leveraging the underlying historical patient data. Detailed experimental evaluations on two sizeable real-world datasets statistically validate the utility of DHC and the deployed engineering efforts demonstrate the real-world impact of the framework.

8. ACKNOWLEDGEMENT

We acknowledge MHS and Microsoft Azure for Research for their generous supports in this research.

References

[1] H. BG et al. Incremental value of clinical data beyond claims data in predicting 30-day outcomes after heart failure hospitalization. *Circ Cardiovasc Qual Outcomes*, 2011.

[2] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[3] S.-H. Cha. Comprehensive survey on distance/similarity measures between probability density functions. *City*, 1(2):1, 2007.

[4] S. Dumais et al. Hierarchical classification of web content. In *SIGIR*, 2000.

[5] J. Friedman et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 2000.

[6] M. Granitzer. Hierarchical text classification using methods from machine learning. 2004.

[7] J. Han and M. Kamber. *Data mining: concepts and techniques*. Morgan Kaufmann, 2006.

[8] S. F. Jencks, M. V. Williams, and E. A. Coleman. Rehospitalizations among patients in the medicare fee-for-service program. *New England Journal of Medicine*, 360(14):1418–1428, 2009.

[9] T. Joachims. Text categorization with support vector machines: Learning with many relevant features, 1998.

[10] R. Kerber. Chimerge: Discretization of numeric attributes. In *Proceedings of the tenth national conference on Artificial intelligence*, pages 123–128. Aaai Press, 1992.

[11] H. M. Krumholz et al. *30-day heart failure readmission measure methodology. Report prepared for the Centers for Medicare & Medicaid Services.*

[12] P. E. Krumholz HM. REadmission after hospitalization for congestive heart failure among medicare beneficiaries. *Archives of Internal Medicine*, 157(1):99–04, Jan. 1997.

[13] M. W. KurzyÅĎski. On the multistage bayes classifier. *Pattern Recognition*, 21(4):355 – 365, 1988.

[14] U. Libal. Multistage pattern recognition of signals represented in wavelet bases with reject option. In *MMAR*, 2012.

[15] V. Rao et al. Readmissions score as a service (raas). 2014.

[16] C. v. Walraven et al. Proportion of hospital readmissions deemed avoidable: a systematic review. *Canadian Medical Association Journal*, 2011.

[17] Y. Yang and X. Liu. A re-examination of text categorization methods. pages 42–49. ACM Press, 1999.

[18] K. Zolfaghar et al. Big data solutions for predicting risk-of-readmission for congestive heart failure patients. In *IEEE Bigdata*, 2013.

[19] K. Zolfaghar et al. Exploring preprocessing techniques for prediction of risk of readmission for congestive heart failure patients. In *DMH Workshop*, 2013.

[20] K. Zolfaghar et al. Predicting risk-of-readmission for congestive heart failure patients: A multi-layer approach. *CoRR*, abs/1306.2094, 2013.

[21] K. Zolfaghar et al. Risk-o-meter: an intelligent clinical risk calculator. In *KDD*, 2013.

Chapter 3

POPULATION COST PREDICTION ON PUBLIC HEALTHCARE DATASETS

Population Cost Prediction on Public Healthcare Datasets

Shanu Sushmita
Center for Data Science
Institute of Technology
University of Washington, Tacoma
sshanu@uw.edu

Stacey Newman
Center for Data Science
Institute of Technology
University of Washington, Tacoma
newmsc8@uw.edu

James Marquardt
Center for Data Science
Institute of Technology
University of Washington, Tacoma
jamarq@uw.edu

Prabhu Ram
Virendra Prasad
Edifecs, Bellevue
[prabhu.ram, virendra.prasad]@edifecs.com

Martine De Cock
Center for Data Science
Institute of Technology
University of Washington, Tacoma
mdecock@uw.edu

Ankur Teredesai
Center for Data Science
Institute of Technology
University of Washington, Tacoma
ankurt@uw.edu

ABSTRACT

The increasing availability of digital health records should ideally improve accountability in healthcare. In this context, the study of predictive modeling of healthcare costs forms a foundation for accountable care, at both population and individual patient-level care. In this research we use machine learning algorithms for accurate predictions of healthcare costs on publicly available claims and survey data. Specifically, we investigate the use of the regression trees, M5 model trees and random forest, to predict healthcare costs of individual patients given their prior medical (and cost) history. Overall, three observations showcase the utility of our research: (a) prior healthcare cost alone can be a good indicator for future healthcare cost, (b) M5 model tree technique led to very accurate future healthcare cost prediction, and (c) although state-of-the-art machine learning algorithms are also limited by skewed cost distributions in healthcare, for a large fraction (75%) of population, we were able to predict with higher accuracy using these algorithms. In particular, using M5 model trees we were able to accurately predict costs within less than $125 for 75% of the population when compared to prior techniques. Since models for predicting healthcare costs are often used to ascertain overall population health, our work is useful to evaluate future costs for large segments of disease populations with reasonably low error as demonstrated in our results on real-world publicly available datasets.

1. INTRODUCTION

Increasing cost of healthcare continues to be one of the world's biggest challenges. According to *The Commonwealth Fund*[1], healthcare costs for 2013 in the United States were highest across the world ($8,508 per-capita). Yet, amongst comparable nations, the United States ranks lowest in terms of quality of care [6]. The problem is not unique to the United States though: from the price of medications and the cost of hospital stays to physicians' fees and medical tests, healthcare costs around the world are skyrocketing. Much of this can be attributed to wasteful spending caused by ineffective drugs and duplicate procedures and paperwork, as well as missed disease-prevention opportunities[2] [5]. With a goal of changing this, healthcare reform policies are currently underway, promoting initiatives for managing the overall health of a population while keeping costs manageable[3].

Accurate prediction of healthcare cost is of immense importance to improve accountability in care. The study of healthcare cost analyses is often directed at getting the most accurate estimate of the mean costs of treating the disease, or identifying the patients'/structure characteristics influencing costs and getting an estimate of expected costs [8]. This study is focused on the latter. Statistical methods for estimating the expected healthcare cost have been largely investigated [2, 7], but are still inconclusive and have several open challenges [8]. Furthermore, previous efforts in the literature for algorithmic prediction of healthcare costs are dominated by linear regression and by rule-based approaches, which require a lot of domain knowledge. But more recently, machine learning approaches like clustering and classification are also being explored for this purpose [1, 11].

Healthcare cost prediction is a challenging problem from the data mining perspective as well. It is well recognized that statistical analysis of healthcare cost data poses a number of difficulties [16]. Cost data is characterized by highly non-Gaussian distributions, which poses interesting challenges for prediction and comparison goals. Data on medical expenditures or costs of treatment typically feature a

DH '15 May 18 - 20, 2015, Florence, Italy

http://dx.doi.org/10.1145/2750511.2750521.

[1] http://www.commonwealthfund.org/publications/fund-reports/2014/jun/mirror-mirror
[2] http://theinstitute.ieee.org/technology-focus/technology-topic/better-health-care-through-data
[3] http://www.hhs.gov/healthcare/facts/timeline

spike at zero and a strongly skewed distribution with a heavy right-hand tail [9]. Another, important challenge is to leverage existing large and varied claims, clinical, and survey data to estimate future healthcare costs, and to take measures in care-management that reduce such costs while improving overall population health.

The goal of this research is to explore machine learning algorithms to provide powerful tools for accurate predictions of healthcare costs. In this paper, we investigate the use of three state-of-the-art machine learning algorithms – regression tree, M5 model tree and random forest, to predict the healthcare costs of individual patients based on data about their previous medical and cost history. To the best of our knowledge, the utility of these algorithms in healthcare costs prediction has not been investigated. We train and test all algorithms on two separate datasets, more specifically a claims (SID) and a survey (MEPS) dataset. Furthermore, we train separate prediction models for four future cost scenarios – three months, six months, nine months and twelve months respectively. That is, given the past, we predict the healthcare cost for the coming three months, six months and so on. We discuss more on this in Section 4. To summarize, the presented research makes the following contributions:

1. Empirical evaluation of regression tree, M5 model tree and random forest to provide powerful tools for accurate predictions of healthcare costs.

2. Empirical evaluation of the regression tree, M5 model tree and random forest on two kinds of datasets – claims and survey data.

3. Empirical evaluation of the regression tree, M5 model tree and random forest for four different future scenarios – three months, six months, nine months and twelve months.

The rest of the paper is organized as follows: In Section 2 we discuss the related background. Two datasets (SID and MEPS) are described in Section 3. Four future cost prediction scenarios are described in Section 4. The three algorithms that we investigate in this study are explained in Section 5. The performance of these algorithms is discussed in Section 6. In Section 7, we show our online cost prediction service where these models are currently deployed. Finally, in Section 8 we conclude with our overall findings.

2. RELATED WORK

Methods: Previously proposed cost prediction models often used rule based methods and multiple linear regression (MLR) models. The challenge with the rule based methods (e.g. [10]) is that they require a lot of domain knowledge, which is not easily available and is often expensive. MLR models are powerful tools for capturing the relationships between the exploratory variables and the dependent variable, but, working with several independent variables often causes the multicollinearity problem, which is caused by the presence of significant correlations among the predictors [18]. In addition, their performance is challenged by the skewed healthcare data. Healthcare cost data typically feature a spike at zero, and a strongly skewed distribution with a heavy right-hand tail [9]. As a result, the prediction models are posed with the challenge of an extreme value situation. It is known that regression models are sensitive to extreme values and likely to be inefficient in small to medium sample sizes if the underlying distribution is not normal [16]. In the past, several advanced statistical methods have been proposed to accommodate the skewness observed in healthcare data, such as General Linear Models (GLM) [13], mixture models based on mixtures of parametric models [17], Markov models [15] etc. For a comprehensive comparison of previously proposed statistical methods for healthcare cost prediction, we refer to the review paper by Mihaylova et al. [16].

The development of accurate healthcare cost prediction models using machine learning methods has been more recent. Bertsimas et al. [1] utilize classification tree and clustering algorithms to provide predictions of healthcare costs in the third year by applying data mining methods to medical and cost data from the first two years. While Lahiri et al. [11], investigate classification algorithms to predict whether an individual is going to incur higher or lower healthcare expenditure. In this paper, we investigate three additional state-of-the-art machine learning algorithms – regression tree, M5 model tree and random forest. To the best of our knowledge, their utility for the healthcare costs prediction problem has not been investigated before.

Evaluation: In previous studies, the models would often estimate the mean cost of the given sampling distribution with a certain confidence interval. Estimation methods utilize observed data as inputs, and do not consider new observations (i.e., observations not in the sample). As a result in these studies, only in-sample data were used to report predictive performance of the methods. In addition, because the relations found by chance in the estimation sample will not replicate, the predictive performance of the model often deteriorates significantly when applied to new data [2]. Therefore, it is more appropriate to express the predictive performance of a method based on out-of-sample experiments (that is, use data that the method has not seen) [1]. In this paper, we divide the data into three parts: a training sample, a validation sample and a testing sample. We measured the performance of the prediction algorithms using the root mean square error (RMSE), mean absolute error (MAE), and the prediction error quantiles. These measures are discussed in detail in Section 6.

Data: The underlying data for building the predictive models often come from claims data, clinical data and/or self-reported survey data (i.e., questionnaires). These datasets are sometimes used separately (e.g., CDPS[4] uses claims data, or PRA[5] uses self-reported data) or as a combination of one or more datasets (e.g., Dorr's algorithm from Care Management Plus[6] uses claims and questionnaire data). Although the predictive power of claims data is often challenged, its utility has been established through several dedicated studies [1, 20]. Furthermore, in the context of building a healthcare cost prediction system for individuals, being able to leverage claims and survey data is beneficial since often the privacy concerns associated with clinical data (such as lab results, vitals, etc.) are far more constrained than those associated with claims and survey data. In this study, we separately train and test our algorithms on claims (SID) as well as survey data (MEPS).

[4] http://cdps.ucsd.edu
[5] https://www.highmarkblueshield.com/pdf_file/ger_binder/pra-overview.pdf
[6] http://caremanagementplus.org

3. DATASET AND FEATURES

In this section we describe the two datasets we use in our study, namely SID and MEPS. In addition, we provide information on the feature set, pre-processing steps, and descriptive statistics of these datasets.

3.1 SID Dataset – Claims Data

The Healthcare Cost and Utilization Project (HCUP) provides Washington State Inpatient Database (SID). In this study we use data from the years 2009 to 2012. The dataset is comprised of a combination of clinical and claims data, collected through a partnership between HCUP and the Washington State Department of Health. This dataset contains several feature sets – diagnoses, procedures, admission charges, comorbidities, revenue codes for specific services, hospital specific data, and injury descriptions. A particularly useful feature of this dataset is the presence of a unique patients' identifier that allows the admissions to be tracked across a given year. More detailed information about the data and its features is available on the HCUP-SID website[7].

In order to build predictive models, we focus on a set of features described as useful in [1]. These include grouper codes for diagnoses and procedures, utilization code units, revenue codes, comorbidities, number of chronic conditions, age, gender, and race. A summary of the feature set used in this study is shown in Table 1.

Variable Number	Description
1-3	Demographics
4-6	Patients' history
7-271	Diagnosis Groups
272-504	Procedures
505-816	Revenue Codes
817-845	Comorbidities
846	Previous Year cost

Table 1: Summary of the features used from the SID dataset.

During the pre-processing stage, the raw SID data was transformed to the individual level from the admission level. Every row in the raw SID data corresponds to a single admission (in the hospital) and its associated information (cost, diagnosis, length of stay, etc.). It is possible that an individual may have several hospital admissions in the given training period, thus leading to several rows in the data. To predict the future healthcare cost of an individual, we use medical and cost information of previous (all) admissions of that individual. In order to transform data from admission to individual level, we summed the values over multiple admissions for diagnosis, procedure, utilization, and cost variables for an individual. For the comorbidity variables, a logical OR was applied. For demographic variables, the value present in the future time window was retained.

In Table 2 we provide descriptive statistics of the SID data after pre-processing. It can be seen that the total number of unique beneficiaries is very large – over 1.5 million. Overall, there are more admissions of female ($\approx 60\%$) than male ($\approx 40\%$) patients. The average healthcare cost of an individual is approximately \$50,000, reflecting the issue of increase in the healthcare cost (also discussed in Section 1).

[7]http://www.hcup-us.ahrq.gov/sidoverview.jsp

Statistics	2009-2012
Unique Beneficiaries	1,884,223
Mean Cost (in dollars)	46,598.35
Mean Age (SD)	45.01 (27.88)
Males	39.71%
Females	60.29%

Table 2: Descriptive statistics from the SID data.

3.2 MEPS Dataset – Survey Data

The Medical Expenditure Panel Survey (MEPS) dataset consists of data extracted from responses to panel surveys given to households and their employers, medical providers, and insurance providers over two year periods[8]. The survey is intended to be representative of the national civilian non-institutionalized population of the United States. Data is available for two year periods from 1996 to 2012. This study utilizes the data from 2004-2012. Data is collected from households[9] for each collection period in a series of five interviews. From those respondents, a sample of their medical providers is contacted in order to provide more accurate medical information. For the purpose of this study, we focus our efforts on the data available in the public MEPS dataset. These are primarily demographic, diagnosis, and care provider outcome based variables. The summary of the feature set used in this study is shown in Table 3.

Variable Number	Description
1-226	Diagnosis Group
227	Previous Year Cost
228-231	Demographics
232-235	Patients' History

Table 3: Summary of the features used from the MEPS dataset.

For the MEPS dataset, we aggregate the data grouping by beneficiary. The feature vector for each beneficiary becomes the number of times a diagnosis was reported during the first year of the survey, their demographics, and their previous and future costs.

Statistics	2004-2012
Unique Beneficiaries	128,312
Households	43,490
Mean Cost (in dollars)	6,518
Mean Age (SD)	34.34 (22.92)
Males	23.09%
Females	9.08%
Unspecified Gender	67.83%

Table 4: Initial statistics from the MEPS data.

Table 4 shows initial statistics about the MEPS data. On an average there are three members per family (households). When compared to the SID dataset, the number of features are low in the MEPS data (see Tables 1 and 3), thus suggesting richer and detailed level of information available in the SID dataset. The average cost (mean cost) of a household

[8]http://meps.ahrq.gov/mepsweb/

[9]Household here means all members of a house (family)

is much lower when compared to the SID data average cost (Table 2). This is expected because SID data is exclusively about patients who visited hospitals, and hospital visit cost are often higher. Another difference between SID and MEPS data is the distribution of male and female. The SID data shows more balance in terms of gender distribution, whereas the distribution is not very clear in the MEPS data due to a large number of unspecified genders.

In summary, the SID data is specifically about hospital admissions (in-patients), while MEPS data contains data about a broader range of medical events (not necessarily in-patient). This is an interesting advantage of the MEPS data over the SID data, and an interesting aspect of our study. On the other hand, the MEPS data is noisier than the SID data because of the way it is collected.

4. PROBLEM DESCRIPTION

The goal of this research is to predict the future healthcare cost of individuals based on their past medical and cost information. More formally, we treat the cost prediction problem as a supervised machine learning problem. The input consists of an attribute vector $\mathbf{x} = (x_1, x_2, x_3.....x_p)$ that contains all available data about the individual, including general demographics such as age and gender, as well as specific clinical and claims data (including cost) for the training period. The output attribute that we aim to predict is the cost y for the result period.

We define *four* future scenarios for predicting cost in this study, that is, predicting future cost for three months, six months, nine months and twelve months. The algorithms (described in Section 5) are trained and tested for the following four scenarios:

P1: We use the history (medical, demographic and cost) of the first three months to predict the cost of the following nine months. Patients with at least one admit in the previous three months were used for the experiments.

P2: We use the history (medical, demographic and cost) of the first six months to predict the cost of the following six months. Patients with at least one admit in the previous six months were used for the experiments.

P3: We use the history (medical, demographic and cost) of the first nine months to predict the cost of the following three months. Patients with at least one admit in the previous nine months were used for the experiments.

P4: We use the history (medical, demographic and cost) of the first twelve months to predict the cost of the following year (twelve months). Patients with at least one admit in the previous year were used for the experiments.

The problem scenarios **P1, P2,** and **P3** are trained and tested using the SID dataset, while for **P4** scenario, we used the MEPS dataset only. This is because, a major downside of the publically available SID dataset is that it is not possible to track individuals across multiple years. Therefore, learning from the previous year to predict healthcare cost for the coming year was not possible with the SID data as in the fourth scenario **P4**. To overcome this limitation, we performed experiments for the fourth scenario using the MEPS dataset. Furthermore, to derive meaningful comparisons between models trained using SID data and MEPS data, we used same features when available in both datasets, or similar features (capturing similar information) in the absence of identical features.

It should be noted that the goal of this research is neither to compare the utility of the SID and MEPS dataset, nor to compare the performance of algorithms on different datasets. Instead, the objective is to perform an empirical evaluation of regression tree, M5 model tree and random forests for the purpose of healthcare cost prediction. These algorithms are trained using the SID and MEPS data depending on the problem scenario at hand.

5. METHODS

In this section we describe the machine learning algorithms including the four baseline models used in this study.

5.1 Baseline Algorithms

1. **Average Baseline:** For this baseline model, we calculate the mean μ of all the individuals' cost within the training set for the training period. The mean (μ) is then multiplied by a factor (k) to make the prediction for all individuals in the test set. For **P1**, the predicted cost = $3 \times \mu$, for **P2**, the predicted cost = $1 \times \mu$, for **P3**, the predicted cost = $0.33 \times \mu$, and for **P4**, the predicted cost = $1 \times \mu$.

2. **Previous Cost Regression (PCR):** For this baseline, a linear regression model is fitted using only the previous cost during the training period as a predictor variable. This model is then used to predict the cost for the testing period.

3. **Multiple Linear Regression (MLR):** We use a multiple linear regression model to predict the cost using a p - dimensional vector of predictive variables. The difference between PCR and MLR is that all features (as shown Table 1 and 3) from SID and MEPS data were used to train the MLR models, while only 'cost' variable was used in the PCR models. We use MLR as an additional baseline due to its extensive use in the literature of healthcare cost prediction.

4. **Generalized Linear Model (GLM):** GLM is a generalization of ordinary least squares regression that relaxes the assumption that the distribution of the response variable be normal[10]. A GLM will consist of a linear predictor (as in ordinary least squares linear regression), a link function that describes how the mean depends on the linear predictor, and a variance function that describes how the variance depends on the mean. For our experiments with GLMs, we assume a Poisson distribution and select the log link function.

5.2 Regression Tree (RTree)

When the data has lots of features which interact in a nonlinear way, assembling a single global model (multiple linear regression) can be very difficult, and confusing. An alternative approach to nonlinear regression is to partition the space into smaller regions, where the interactions are

[10]http://en.wikipedia.org/wiki/Generalized_linear_model

more manageable. The goal of a regression tree is to predict a response y (cost in our case) from inputs $x_1, x_2, ...x_p$. This is done by growing a binary tree. At each internal node in the tree, a test is applied to one of the inputs, for example x_i. Depending on the outcome of the test, the left or the right sub-branch of the tree is then selected. Eventually a leaf node is reached, where the prediction is made. For this study, we used an implementation of classification and regression tree algorithm (CART)[4] in R. The minimum deviance (mean squared error) was used as the test parameter for proceeding with a new split. That is, adding a node (feature selection) should reduce the error by at least a certain amount. We tested the performance of regression trees using different complexity parameters (cp=0.01, 0.001, 0.0005). In Table 5 we report the best performing tree with cp set to 0.0005 value.

5.3 Random Forest Regression (RFR)

Random forest regression is an ensemble learning method that operates by constructing a multitude of regression trees at training time and outputting the mean prediction of the individual trees for new observations. Each tree is constructed using a random sample of the row (observation) and column (feature) space of the original dataset. This has the effect of correcting the tendency of individual regression trees to overfit the training data [3]. For this work we utilize the implementation of random forests in the R, and grow all regression trees without pruning[12].

5.4 M5 Model Tree (M5)

M5 model trees are a generalization of the CART model. The structure of a M5 model tree follows that of decision tree, but has multiple linear regression models at the leaf nodes, making the model a combination of piecewise linear functions. The algorithm for the training of a model tree breaks the input space of the training data through a recursive partitioning process similar to the one used in CART. After partitioning, linear regression models can be fit on the leaf nodes, making the resulting regression model locally accurate [19].

All the models, including baseline models and three machine learning algorithms are trained using R.

6. RESULTS

We measure the performance of the prediction algorithms using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), with a lower error indicating a better performance. We test our models using 10-fold cross validation. Traditionally, R^2 or adjusted R^2 have been used to evaluate healthcare cost prediction models, but there are some drawbacks to their use. While R^2 does measure how well a model fits the training data, the metric is not a true indicator of how well a model will predict unseen observations [1]. Therefore, we use MAE and RMSE in this study. Both metrics measure the predictive quality of a model, with RMSE being more sensitive to the outliers. Formally, MAE is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|,$$

and RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}},$$

where $\hat{y}_i$ is the predicted value, y_i the true value, and n is the number of observations in the sample.

In addition to the single value metrics described, we measure the performance of the algorithms by examining the absolute prediction error distribution. As models for predicting healthcare costs are often used to predict the overall healthcare cost of a population, it is useful to evaluate whether large segments of a test set are predicted with reasonably low error.

The RMSE and MAE results for the four future scenarios **P1, P2, P3** and **P4** are shown in Table 5. In addition to results for the regression trees, M5 model tree and random forest, we include results for all four baseline algorithms – average, previous cost, linear regression and GLM (described in Section 5).

Algorithm	RMSE ($)	MAE ($)
P1 (SID Data)		
AB	127,156	115,875
PCR	64,414	27,669
MLR	92,064	29,007
GLM	152,779	81,497
RTree	67,664	26,480
M5	79,790	20,715
RFR (n =50)	66,340	25,655
P2 (SID Data)		
AB	72,367	51,970
PCR	53,405	19,633
MLR	126,062	22,448
GLM	157,785	79,443
RTree	67,344	19,057
M5	77,242	14,607
RFR (n =50)	52,581	17,563
P3 (SID Data)		
AB	69,239	23,854
PCR	61,259	12,710
MLR	82,361	18,652
GLM	233,733	80,485
RTree	72,888	11,671
M5	68,864	7,647
RFR (n = 50)	66,066	11,293
P4 (MEPS Data)		
AB	36,328	12,474
PCR	34,715	11,387
MLR	33,631	10,597
GLM	37,397	8,886
RTree	34,434	10,568
M5	35,476	8,112
RFR (n= 50)	33,618	10,060

Table 5: Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) using SID and MEPS data. Where, AB = Average Baseline, PCB = Previous Cost Regression, MLR = Multiple Linear Regression Baseline, GLM = Generalized Linear Models, RTree = Regression Tree, M5 = M5 Model Tree, and RFR = Random Forest. For RFR, n is the number of trees, and for GLM, we assume a Poisson distribution and use the log link function.

As can be seen in Table 5, for future scenario **P1**, all three models outperformed (lower prediction errors) the baseline models in terms of MAE. Among them, M5 model tree had

the lowest MAE. With respect to the RMSE values, all three algorithms performed better than average, multiple linear regression and GLM baselines, but, none could outperform the *previous cost* baseline (PCR).

For future scenario **P2**, all three models performed better than the baseline models with respect to MAE values, and the M5 model tree was the best performing model again. With respect to RMSE, random forest was the best performing model, but, regression and M5 model tree could not outperform the *previous cost* baseline (PCR).

In case of **P3** scenario, mean absolute prediction errors (MAE) by M5 model trees were very low when compared to all baseline models. Regression tree and random forest were the next best performing models. As seen in previous scenarios (**P1, P2**), RMSE values were again very high for all models when compared to MAE values, GLM being the worst model. None of the machine learning algorithms (regression tree, random forest and M5 model trees) could outperform the *previous cost* baseline (PCR) when compared against RMSE values.

Finally, for the **P4** scenario, M5 model tree again had the lowest MAE values, but the performance of the GLM baseline was also comparable. Random forest had the lowest RMSE values but the difference was not large when compared to other algorithms. RMSE values of the regression tree, M5 model tree and all four baselines were quite close (similar error values). It should be noted that results for **P4** were obtained using MEPS data, while results for **P1, P2** and **P3** were obtained using SID data.

Overall, three key observations can be made from the performance results shown in Table 5. First, *previous cost regression (PCR)* is a strong baseline, and therefore previous healthcare cost alone can be a good indicator for future healthcare cost. Second, among the three machine learning algorithms (regression tree, random forest and M5 model tree), M5 model tree consistently performed best and achieved a substantially lower MAE than strong PCR baseline for predicting future healthcare cost in all scenarios. Third, while all three machine learning algorithms (regression tree, random forest and M5 model tree) outperform the PCR baseline with respect to MAE, their comparative performance with respect to RMSE is far less convincing. This might indicate that while the three algorithms do really well on average, they are prone to higher variance and make much larger errors than PCR baseline.

In order to further investigate the error values (in Table 5), we looked at the quartiles of the absolute error (deviation from the actual cost) distribution for all the algorithms. The error distribution results are shown in Figure 1, 2, 3 and 4. The x axis shows the proportion of the population, and the y-axis shows the error distribution in dollar amount. For visualization and discussion purposes, we include error distribution upto 75 percentile only. Complete distribution (upto 100 percentile) can be seen in the Table[11] link provided.

For all four scenarios, it can be seen that for 75% of the test data, the error on the predicted cost by the models was less than $30,000. In particular, for the **P3** scenario, the error is as low as $125 using M5 model. This result is promising because for a large fraction of the population (75%), we were able to predict with higher accuracy using these algorithms. This is an interesting observation because

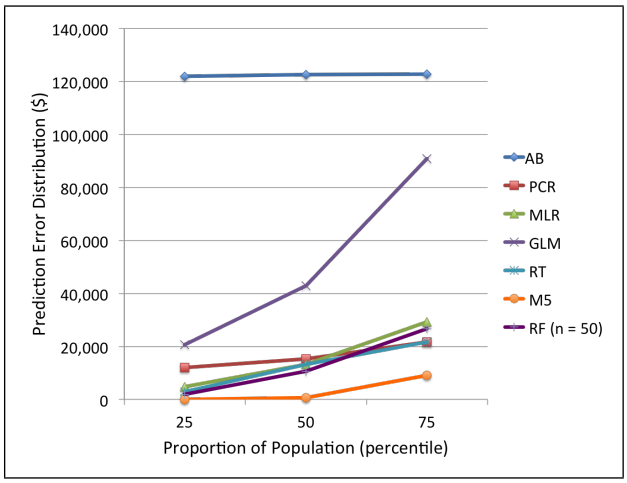

Figure 1: Absolute Error distribution summary for models in **P1** scenario. The x axis shows the proportion of the population, and the y-axis shows the error distribution in dollar amount. Here, AB = Average Baseline, PCR = Previous Cost Regression (Baseline), MLR = Multiple Linear Regression (Baseline), GLM = Generalized Linear Models (Baseline), RTree = Regression Tree, M5 = M5 Model Tree, and RFR = Random Forest Regression. For RFR, n is the number of trees, and for GLM, we assume a Poisson distribution and use the log link function.

from an accountable care organizations (ACOs) perceptive, predicting aggregated cost for population can be very useful. For the **P2, P3** and **P4** scenario, the error values were between $0 - $7,000 dollars for 50% of test data (See Figure 2, 3 and 4). Although the results are promising, but they also reflect the challenges of modeling skewed distribution of healthcare cost data because of large overall RMSE and MAE values (also discussed in Section 2), and that modern machine learning algorithms are not immune to this issue. It might be possible to improve the performance of these algorithms by removing extreme values. This could be an interesting problem to investigate in the future.

7. HEALTHCARE SCALABLE COST PREDICTION ENGINE

The models described in previous sections are deployed on HealthSCOPE (Healthcare Scalable COst Prediction Engine)[12], which is a framework for exploring historical and present day healthcare costs as well as for predicting future costs. HealthSCOPE can be used by individuals to estimate their healthcare costs in the coming year. In addition, HealthSCOPE supports a population based view for actuaries and insurers who want to estimate the future costs of a population based on historical claims data, a typical scenario for accountable care organizations (ACOs).

Using our interactive data mining framework, users can view claims (sample files are provided for demo purposes), use HealthSCOPE to predict costs for the upcoming year, interactively select from a set of possible medical conditions, understand the factors that contribute to the cost, and compare costs against historical averages (See Figure

[11]http://tinyurl.com/ErrorDistributionTable

[12]http://healthscope.cloudapp.net/hscope-dev/aco/

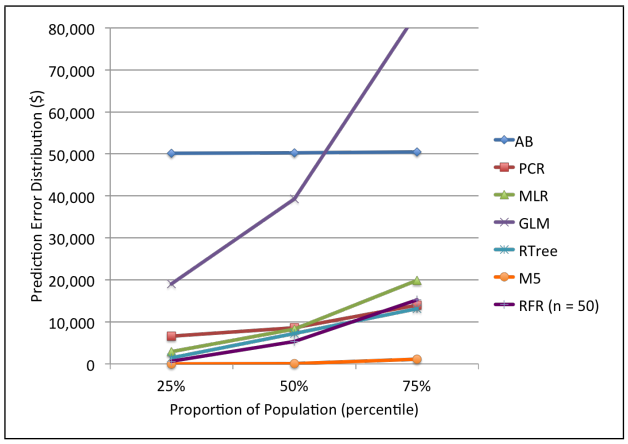

Figure 2: Absolute Error distribution summary for models in **P2** scenario. The x axis shows the proportion of the population, and the y-axis shows the error distribution in dollar amount. Here, AB = Average Baseline, PCR = Previous Cost Regression (Baseline), MLR = Multiple Linear Regression (Baseline), GLM = Generalized Linear Models (Baseline), RTree = Regression Tree, M5 = M5 Model Tree, and RFR = Random Forest Regression. For RFR, n is the number of trees, and for GLM, we assume a Poisson distribution and use the log link function.

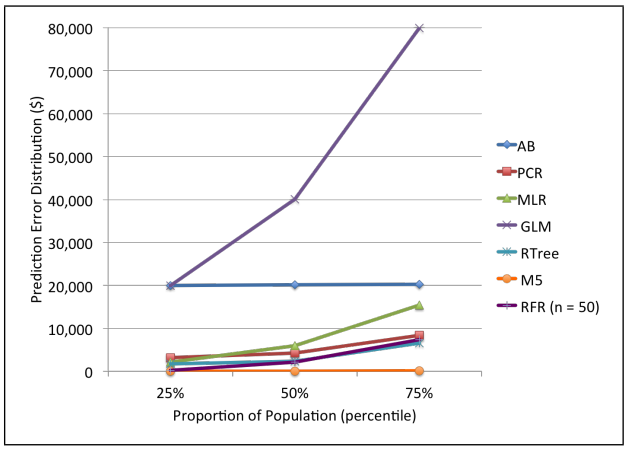

Figure 3: Absolute Error distribution summary for models in **P3** scenario. The x axis shows the proportion of the population, and the y-axis shows the error distribution in dollar amount. Here, AB = Average Baseline, PCR = Previous Cost Regression (Baseline), MLR = Multiple Linear Regression (Baseline), GLM = Generalized Linear Models (Baseline), RTree = Regression Tree, M5 = M5 Model Tree, and RFR = Random Forest Regression. For RFR, n is the number of trees, and for GLM, we assume a Poisson distribution and use the log link function.

5). The back-end system contains cloud based prediction services hosted on the Microsoft Azure infrastructure that allow the easy deployment of models encoded in Predictive Model Markup Language (PMML) and trained using either Spark MLLib or various non-distributed environments. More details about the underlying framework and individual modules can found here [14].

8. CONCLUSION

Accurate prediction of healthcare cost is of immense importance to improve accountability in care. As a result, the analysis of healthcare costs has become an important part of both experimental and epidemiological research. The goal of the presented research was to investigate modern machine learning algorithms for the task of predictions of future healthcare costs. In this paper, we investigated three algorithms – regression tree, M5 model tree and random forest using claims and survey data. In addition, empirical evaluation of these algorithms for four different future cost prediction scenarios – three months, six months, nine months and twelve months were also performed. Overall, three key observations were made during this study. First, previous healthcare cost alone can be a good indicator for future healthcare cost. Second, M5 model tree shows potential for solving future healthcare cost prediction problems. Third, state-of-the-art machine learning algorithms are also limited by the skewed distribution of healthcare cost data. However, for a large fraction (75%) of the population, we were able to predict with higher accuracy using these algorithms. As models for predicting healthcare costs is often used to predict the overall healthcare cost of a population, it is useful to evaluate whether large segments of a test set are predicted with reasonably low error. As continuation of this analysis, we plan to take a deeper dive into the data and explore ways to improve the performance of algorithms, may be through feature selection, or by filtering the outliers that are several orders larger than the other samples when calculating the errors (in particular RMSE). In addition, identifying (automatically) which are the difficult cases (individuals, sub-population, etc) to predict accurately (have large errors), and build dedicated models for those cases.

Acknowledgements

We would like to thank Dr. George Wu, David Hazel and Senjuti Roy Basu for their helpful suggestions and feedback. We would also like to thank the Azure For Research[13] program from Microsoft Research Connections for the compute resources grant enabling us to use Azure infrastructure for this research, and Edifecs Inc., for their generous support to the Center for Data Science, and Zementis for the donation of licenses for their ADAPA software.

9. REFERENCES

[1] D. Bertsimas, M. V. Bjarnadóttir, M. A. Kane, J. C. Kryder, R. Pandey, S. Vempala, and G. Wang. Algorithmic prediction of health-care costs. *Operations Research*, 56(6):1382–1392, 2008.

[2] M. Bilger and W. G. Manning. Measuring overfitting in non-linear models: A new method and an application to health expenditures. *International Journal of Health Economics*, 24(1):75–85, 2015.

[3] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[4] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth Publishing Company, 1984.

[13] http://azure4research.com/

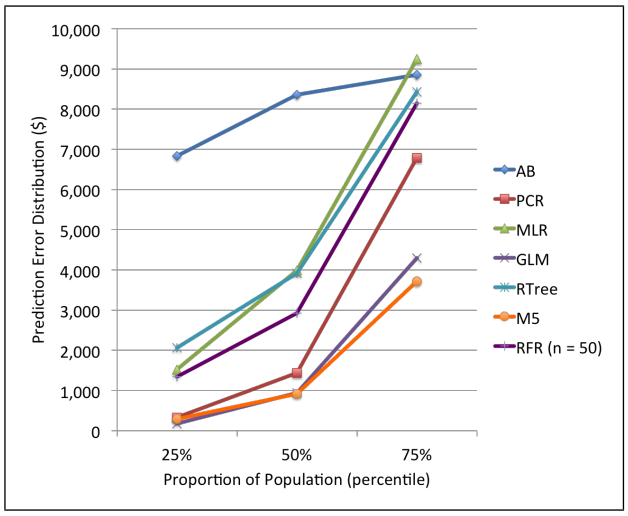

Figure 4: Absolute Error distribution summary for models in **P4** scenario. The x axis shows the proportion of the population, and the y-axis shows the error distribution in dollar amount. Here, AB = Average Baseline, PCR = Previous Cost Regression (Baseline), MLR = Multiple Linear Regression (Baseline), GLM = Generalized Linear Models (Baseline), RTree = Regression Tree, M5 = M5 Model Tree, and RFR = Random Forest Regression. For RFR, n is the number of trees, and for GLM, we assume a Poisson distribution and use the log link function.

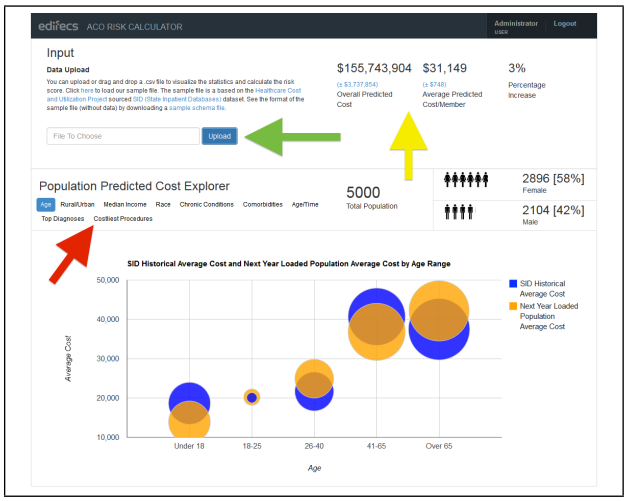

Figure 5: Screenshot showing previous and predicted costs for the group of individuals across different age groups. The green arrow indicates the form for uploading healthcare data to be evaluated, the red arrow indicates navigation options for different population visualizations, and the yellow arrow indicates population cost predictions. In the current visualization, it can be seen that for the individuals over age 65, future healthcare cost (yellow bubble) is predicted to be higher that the previous year cost (blue bubble).

[5] V. Chandola, S. R. Sukumar, and J. C. Schryver. Knowledge discovery from massive healthcare claims data. In *Proceedings of the 19th ACM SIGKDD*, pages 1312–1320, 2013.

[6] N. R. Council and I. of Medicine. *U.S. Health in International Perspective: Shorter Lives, Poorer Health*. The National Academies Press, 2013.

[7] P. Diehr, D. Yanez, A. Ash, M. Hornbrook, and D. Y. Lin1. Methods for analyzing health care utilization and costs. *Annual Review of Public Health*, 20(1):125–44, 2007.

[8] D. Gregori, M. Petrinco, S. Bo, A. Desideri, F. Merletti, and E. Pagano. Regression models for analyzing costs and their determinants in health care: an introductory review. *International Journal of Quality Health Care*, 23(3):331–41, 2011.

[9] A. Jones. Models For Health Care. Technical report, HEDG, c/o Department of Economics, University of York, Jan. 2010.

[10] R. Kronick, T. P. Gilmer, T. Dreyfus, and T. G. Ganiats. Cdps-medicare: The chronic illness and disability payment system modified to predict expenditures for medicare beneficiaries. Technical report, 2002.

[11] C. B. Lahiri and N. Agarwal. Predicting healthcare expenditure increase for an individual from medicare data. In *Proceedings of the ACM SIGKDD Workshop on Health Informatics*, 2014.

[12] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[13] W. G. Manning, A. Basu, and J. Mullahy. Generalized modeling approaches to risk adjustment of skewed outcomes data. *Journal of Health Economics*, 24(3):465–488, 2005.

[14] J. Marquardt, S. Newman, D. Hattarki, R. Srinivasan, S. Sushmita, P. Ram, V. Prasad, D. Hazel, A. Ramesh, M. D. Cock, and A. Teredesai. Healthscope: An interactive distributed data mining framework for scalable prediction of healthcare costs. In *In Proceedings of the IEEE ICDM*, 2014.

[15] A. H. Marshall, B. Shaw, and S. I. McClean. Estimating the costs for a group of geriatric patients using the Coxian phase-type distribution. *Statistics in Medicine*, 26:2716–2729, 2007.

[16] B. Mihaylova, A. Briggs, A. O'Hagan, and S. G. Thompson. Review of statistical methods for analysing healthcare resources and costs. *Health Economics*, 20(8):897–916, 2011.

[17] J. Mullahy. Heterogeneity, excess zeros, and the structure of count data models. *Journal of Applied Econometrics*, 12(3):337–50, 1997.

[18] C. V. PATRICHE, R. G. PIRNAU, and B. ROÅđCA. Comparing linear regression and regression trees for spatial modelling of soil reaction in dobrovĂČĂč basin (eastern romania). *Bulletin UASVM Agriculture*, 68(1):264–271, 2011.

[19] J. R. Quinlan. Learning with continuous classes. In *In Proceedings of AI*, pages 343–348. Adams and Sterling, 1992.

[20] Y. Zhao, A. S. Ash, R. P. Ellis, J. Z. Ayanian, G. C. Pope, B. Bowen, and L. Weyuke. Predicting pharmacy costs and other medical costs using diagnoses and drug claims. *Journal of Medical Care*, 43(1):34–43, 2005.

Chapter 4

HEALTHSCOPE: AN INTERACTIVE DISTRIBUTED DATA MINING FRAMEWORK FOR SCALABLE PREDICTION OF HEALTHCARE COSTS

2014 IEEE International Conference on Data Mining Workshop

HealthSCOPE: An Interactive Distributed Data Mining Framework for Scalable Prediction of Healthcare Costs

James Marquardt*, Stacey Newman*, Deepa Hattarki*, Rajagopalan Srinivasan*, Shanu Sushmita*, Prabhu Ram†, Viren Prasad†, David Hazel*, Archana Ramesh*, Martine De Cock*‡, Ankur Teredesai*
*Center for Data Science, Institute of Technology
University of Washington Tacoma
1900 Commerce Street, Tacoma WA-98402, USA
Email: {jamarq, newmsc8, hattd, velamurr,sshanu, dhazel, aramesh2, mdecock, ankurt}@uw.edu
†Edifecs
2600 116th Avenue Ne #200, Bellevue WA-98004, USA
Email: {prabhu.ram,VirendraP}@edifecs.com
‡Dept. of Applied Mathematics, Computer Science and Statistics
Ghent University
Krijgslaan 281 (S9), 9000 Gent, Belgium
Email: martine.decock@ugent.be

***Abstract*—In this demonstration proposal we describe HealthSCOPE (Healthcare Scalable COst Prediction Engine), a framework for exploring historical and present day healthcare costs as well as for predicting future costs. HealthSCOPE can be used by individuals to estimate their healthcare costs in the coming year. In addition, HealthSCOPE supports a population based view for actuaries and insurers who want to estimate the future costs of a population based on historical claims data, a typical scenario for accountable care organizations (ACOs).**

Using our interactive data mining framework, users can view claims (sample files will be provided), use HealthSCOPE to predict costs for the upcoming year, interactively select from a set of possible medical conditions, understand the factors that contribute to the cost, and compare costs against historical averages. The back-end system contains cloud based prediction services hosted on the Microsoft Azure infrastructure that allow the easy deployment of models encoded in Predictive Model Markup Language (PMML) and trained using either Spark MLLib or various non-distributed environments.

Keywords*—*Healthcare cost prediction, insurance claims data, distributed data mining, Spark, Microsoft Azure, PMML

I. Introduction

Healthcare cost prediction is of immense importance to improve accountability in care. According to the World Health Organization, healthcare costs for 2012 in the United States were highest across the world; both per-capita ($8,233) as well as percentage of the Gross Domestic Product (17.6%). Yet, amongst comparable nations, the United States rank towards the bottom in terms of quality of care (1). With a goal of changing this, healthcare reform policies are currently underway, promoting initiatives for managing the overall health of a population while keeping costs manageable (2). Important challenges for this reform are to leverage existing large and varied clinical and claims datasets to estimate future healthcare costs, and to take measures in care-management that reduce such costs while improving overall population health. Data used for cost prediction models are often voluminous, diverse, and vary significantly over time. As the amount and complexity of data increases, so does the challenge to store, retrieve and manipulate. A scalable healthcare cost prediction system should aim to provide high-availability, fault-tolerance and fast-processing of a large number of claims so that the overall quality of the services is not affected. Handling such diverse, voluminous, and dynamic data is still a challenge for most of the existing healthcare cost prediction algorithms, and to the best of our knowledge a framework providing distributed algorithms and corresponding web based APIs to support individual as well as population level analytics has not been demonstrated.

In this demonstration proposal, we describe HealthSCOPE (*Health*care *S*calable *CO*st *P*rediction *E*ngine) – an interactive predictive modeling framework for the exploration of healthcare costs. HealthSCOPE allows for fast analysis and estimation of costs, as well as ease of predictive model deployment. It is a generic healthcare cost prediction framework that allows individuals to upload their healthcare history and understand costs. HealthSCOPE can make cost predictions for a population of beneficiaries[1]. End users (typically managers at Accountable Care Organizations – ACOs, or benefit managers at insurance companies) can upload basic demographics, recent diagnoses, and comorbidities of a population for the past year and browse through population characteristics, estimates of the the healthcare costs associated with the next year and the factors that contribute most to high costs. This second use case scenario caters to hospitals and insurance providers who are interested in understanding costs associated with a heterogeneous population, given a history of insurance claims for that population.

[1]A beneficiary, in the context of this proposal, is an individual (patient) who receives the benefits of a health insurance contract.

Previous efforts and tools are restricted both in their dependence on linear regression (which cannot handle high dimensional and varied data), and rule based approaches (which require a lot of domain knowledge and on-premise compute infrastructure typically expensive to scale), but more recently, data mining approaches are also being explored for this purpose (3). However, to the best of our knowledge regression trees have not been used for the cost prediction problems. HealthSCOPE benefits from regression trees and exhibits better performance when compared to a linear regression model. Further, we are not aware of any claims based cost prediction as a service implementations (either for individuals or for populations).

HealthSCOPE is powered by Spark[2], which provides fast and parallel processing of large scale data through cluster computing. A useful feature of HealthSCOPE is that it allows consumers, health plan portals, or ACOs to simply upload their own claims data and use scalable implementations of supervised machine learning methods for prediction and exploration purposes. Additionally, HealthSCOPE allows further development and deployment of predictive models trained using Spark MLLib. HealthSCOPE provides the capability to export predictive models trained using Spark MLLib to PMML[3].The rest of the paper is structured as follows: In Section II we present the user interface of HealthSCOPE and how it can be used by the end user. The design of the back-end system is described in Section III followed by demonstration logistics. Related work is addressed as relevant throughout the proposal.

II. HealthSCOPE Features and Demonstration Overview

HealthSCOPE provides a framework for exploring historical and present day healthcare costs, as well as for predicting future costs. HealthSCOPE can be used by individuals to estimate their healthcare costs in the coming year. In addition, HealthSCOPE supports a population based view for actuaries and insurers who want to estimate the future costs of a population based on historical claims data, a typical scenario for accountable care organizations (ACOs). In this proposal, due to space limitations, we describe the ACO user scenario.

A. Population Level View

There is growing interest in reducing overall healthcare costs, and one of the best ways to control medical costs is by improving the overall health of the patient population (4). To this effect, various payment models are being explored. One model that has gained traction is called Accountable Care Organizations or ACOs. An ACO is often a group of physicians and other healthcare organizations such as hospitals who form an entity and work together with Medicare or commercial healthcare insurance companies to provide care to a population of patients. ACOs share both financial and medical responsibility for providing coordinated care to patients in hopes of limiting unnecessary spending. ACOs coordinate care for their patients on an individual basis and reach across medical specialties and care settings. By managing the health of the overall patient population and the health of individual high risk patients, ACOs aim to achieve cost savings.

Fig. 1. Screenshot showing combined and average predicted costs for the group of beneficiaries across different age groups. The green arrow indicates the form for uploading healthcare data to be evaluated, the red arrow indicates navigation options for different population visualizations, and the yellow arrow indicates population cost predictions. In the current visualization, it can be seen that for the beneficiaries over age 65, future healthcare cost (yellow bubble) is predicted to be higher that the previous year cost (blue bubble).

In order to cater to the population level responsibilities of ACOs, HealthSCOPE generates a combination of visualizations and cost predictions deemed valuable for analysis. Upon loading health data relevant to a particular population using the upload button (as marked with a green arrow in the Figure 1), the ACO user will see combined and average predicted costs for the group of beneficiaries being evaluated. This information is highly valuable in gauging the overall cost risk of a particular population. Several charts are also generated to convey what factors contribute to the overall predicted costs. These charts detail the breakdown of costs with respect to groupings of beneficiaries in terms of age, race, income level, and residence location. An example breakdown of predicted cost and historic cost across different age groups can be seen in Figure 1. Such a breakdown across different groups can help to identify groups that may cause an increase in overall healthcare cost in future.

B. Beneficiary Level View

As an extension to their population level exploration, analysis at the beneficiary level is also very valuable. By identifying high-risk beneficiaries, ACOs and providers can focus more attention on specific beneficiaries to improve overall health. In order to facilitate this level of analysis, HealthSCOPE identifies beneficiaries within a population who are forecast to have a particularly large increase in cost in future years. Upon selecting a beneficiary to evaluate more closely, the user is presented with the factors that make the beneficiary in question costly, as seen in Figure 2. In particular, actionable items such as comorbidities are highlighted. As a function to demonstrate how certain interventions will affect

[2]https://spark.apache.org

[3]Predictive Model Markup Language (PMML) provides an open standard for representing data mining models. In this way, models can easily be shared between different applications avoiding proprietary issues and incompatibilities. Currently, all major commercial and open source data mining tools already support PMML.

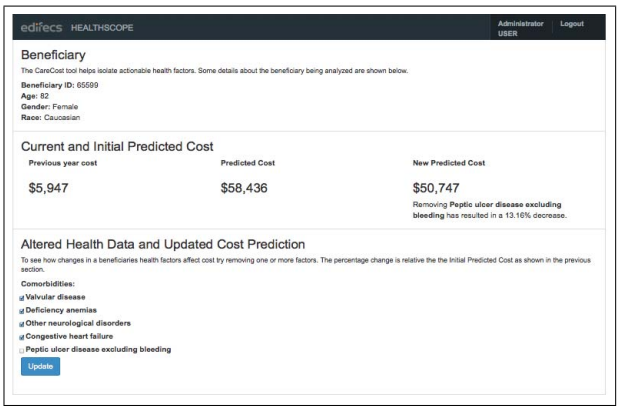

Fig. 2. Screenshot showing the individual level view reached from the population level evaluation.

the beneficiary's forecasted cost, the user is able to remove particular comorbidities from the beneficiary's record. Upon completing these modifications to the record, an updated cost prediction is generated for the beneficiary. For instance, in Figure 2 it can been be seen that the comorbidity "peptic ulcer" is one of the contributing factors in the overall cost increase for the selected beneficiary. The user, can uncheck this comorbidity and get an estimate on how this intervention will affect the beneficiary's forecasted cost. In this case, the new prediction shows a drop of 13% in the overall forecasted cost.

III. HealthSCOPE Design and Internals

The overall architecture of HealthSCOPE is shown in Figure 3. The framework consists of a user interface (UI), a cost prediction API (which is hosted on Microsoft Azure) and a Cost Prediction Engine (CPE). Next, we describe these modules.

a) User Interface (UI): The user interface shown in Figure 1 allows the user to upload data (in csv) with details like age, race, gender, location (through zip code), chronic conditions, and so on. Sample files for the demonstration are provided. After upload the UI provides a predicted overall cost and basic gender breakdown on the right. It also indicates how much of an increase or decrease from total cost for the population one can expect. In addition there are several visualization options for exploring the underlying uploaded data and the cost predictions (see Section II, Figure 1 and 2).

b) Cost Prediction API: The entire service is accessed through Representational State Transfer(REST) APIs using HTTP calls. Data entered by the user is parsed into comma separated values by this module, and is then sent to the model selector. In some user scenarios (e.g, ACO user) this layer also computes the population aggregations and visualization statistics. The Cost Prediction API collects the predicted costs from the cost prediction engine and sends the predicted values back to the user interface.

c) Cost Prediction Engine: We utilize ADAPA (Adaptive Decision and Predictive Analytics)[4] to perform beneficiary scoring. The Cost prediction Engine consists of the following additional sub-modules:

1) *Model Selector:* This sub-module is configurable to send the incoming input variables from the Cost Prediction API to one of the appropriate prediction models available in the Model Bank.
2) *Model Bank:* ADAPA is used to load and deploy predictive models in PMML format for scoring. Currently, there are two prediction models available in the Model Bank: Linear Regression and Regression Trees. Both models are trained using the R statistical computing language, and using MLlib in Spark. For evaluation purposes, the performance and error of each model is compared to a mean baseline, and linear regression model. We use linear regression as an additional baseline due to its extensive use in the literature of healthcare cost prediction. Our models have shown improvement (lower prediction errors) over linear regression as well as average baseline models. In our ongoing research, we continue to explore additional algorithms like weighted KNN, SVM, Naive Bayes, etc. Our future goal is to add more trained models to the model bank collection.
3) *Big Data Stack:* The Big Data stack is powered by Spark, which provides fast and distributed processing of large scale data on a cluster of commodity hardware.

The prediction models in the model bank of HealthSCOPE's Cost Prediction Engine are trained on historical insurance claims data. While healthcare datasets are often used for predicting future healthcare cost, the goal varies from predicting individual cost, to estimating total population healthcare costs (5; 6; 3). The underlying data for building these predictive models often come from claims data, clinical data and/or self-reported data (i.e., questionnaires). These datasets are sometimes used separately (e.g., CDPS[5] which uses claims data, or PRA[6] which uses self-reported data) or as a combination of one or more datasets (e.g., Dorr's algorithm from Care Management Plus[7] uses claim and questionnaire data). Although the predictive power of claims data is often challenged, its utility has been established through several dedicated studies (3; 7). Furthermore, in the context of building a healthcare cost prediction system for individuals, being able to leverage claims data is beneficial since often the privacy concerns associated with clinical data (such as lab results, vitals, etc.) are far more constrained than those associated with claims data.

HealthSCOPE is currently trained on the State Inpatient Database (SID) of the state Washington. This database is part of the family of databases developed for the Healthcare Cost and Utilization Project (HCUP)[8]. The dataset consists of inpatient discharge records from community hospitals in the State of Washington with all-payer, encounter-level information from 2011 to 2012. The data consists of 650,000 records

[4]http://zementis.com/products/adapa/

[5]http://cdps.ucsd.edu

[6]https://www.highmarkblueshield.com/pdf_file/ger_binder/pra-overview.pdf

[7]http://caremanagementplus.org

[8]http://www.hcup-us.ahrq.gov/db/state/siddbdocumentation.jsp

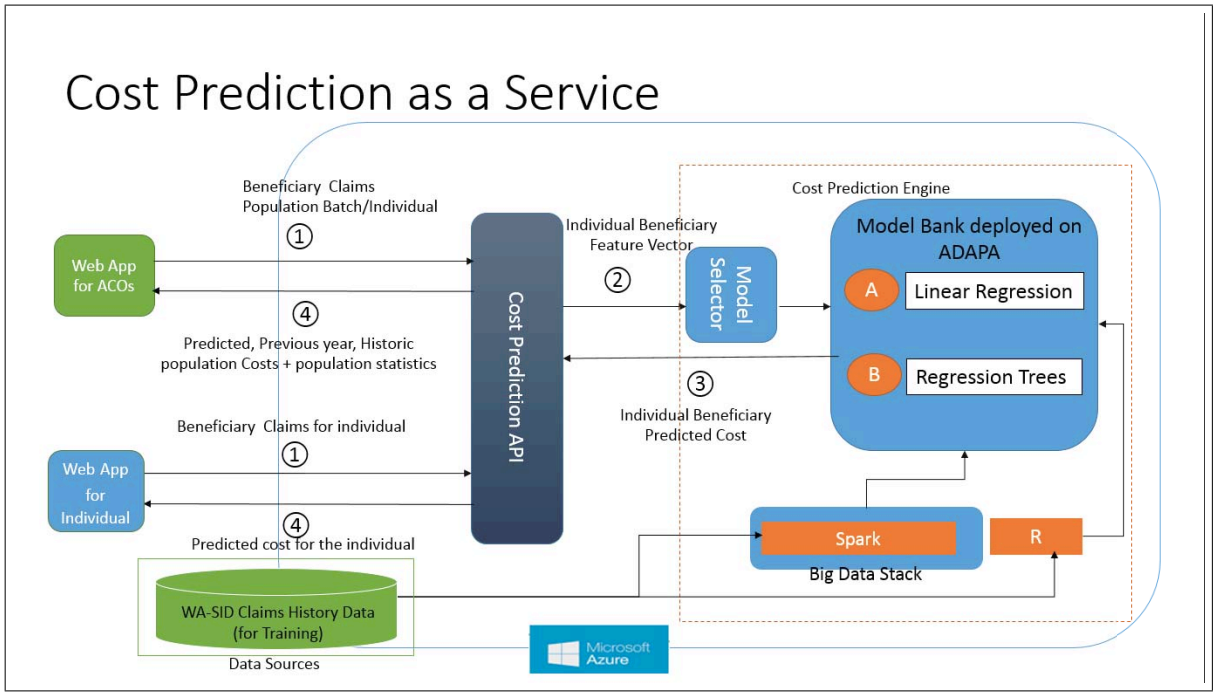

Fig. 3. Overall architecture of HealthSCOPE. A user can enter (upload) data with details like age, race, gender, location (through zip code), and chronic conditions through a Web UI. The data submitted by the user is then sent to the Cost Prediction API. The API parses the data into comma separated values and sends it to the Model Selector. Based on the configuration in the Model Selector the parsed values are sent to one of the prediction models in the Model Bank. The predicted values are sent back through the Cost Prediction API to the user. The whole API is deployed in Microsoft Azure.

per year corresponding to 480,000 beneficiaries with over 1000 attributes. Some of the example attributes are shown in Table I. More detailed information about the data and its features is available on the HCUP-SID website[9].

Attributes
Principal and secondary diagnoses and procedures
Hospital utilization codes and charges
Patient demographics characteristics (e.g., sex, age, and, for some states, race)
Total charges
Length of stay

TABLE I. EXAMPLE ATTRIBUTES PRESENT IN THE SID DATASET.

DEMONSTRATION LOGISTICS

The population and individual level interfaces for our framework can be found at http://tinyurl.com/healthscope-aco and http://tinyurl.com/healthscope-indiv respectively. For the purposes of this conference, we will provide a laptop on which to access the application as well as a projector. We will require access to a table large enough to accommodate both the laptop and the projector, as well as access to power outlets for both.

ACKNOWLEDGEMENTS

We would like to thank Dwaine Trummert for the UI Design, as well as Ila Nejati for their helpful suggestions. We would also like to thank the Azure For Research[10] program from Microsoft Research Connections for the compute resources grant enabling us to use Azure infrastructure for this research, Edifecs Inc., for their generous support to the Center for Data Science, and ADAPA for the donation of licenses for their software.

[9]http://www.hcup-us.ahrq.gov/sidoverview.jsp
[10]http://azure4research.com/

REFERENCES

[1] S. Woolf and L. Aron, Eds., *U.S. health in international perspective: Shorter lives, poorer health*. National Academies Press (US), 2013.
[2] "Key features of the affordable care act," http://www.hhs.gov/healthcare/facts/timeline, accessed on June 25, 2014.
[3] D. Bertsimas, M. V. Bjarnadóttir, M. A. Kane, J. C. Kryder, R. Pandey, S. Vempala, and G. Wang, "Algorithmic prediction of health-care costs," *Operations Research*, vol. 56, no. 6, pp. 1382–1392, 2008.
[4] "Return on investments in health," http://www.rwjf.org/content/dam/farm/reports/issue_briefs/2013/rwjf72446, accessed on June 27, 2014.
[5] J. L. Meyers, S. Parasuraman, K. F. Bell, J. P. Graham, and S. D. Candrilli, "The high-cost, type 2 diabetes mellitus patient: an analysis of managed care administrative data," *Archives of Public Health*, vol. 72, no. 1, p. 6, 2014.
[6] M. Seid, J. W. Varni, D. Segall, and P. S. Kurtin, "Health-related quality of life as a predictor of pediatric healthcare costs: a two-year prospective cohort analysis," *Health and quality of life outcomes*, vol. 2, no. 1, p. 48, 2004.
[7] Y. Zhao, A. Ash, R. Ellis, J. Ayanian, G. Pope, B. Bowen, and L. Weyuker, "Predicting pharmacy costs and other medical costs using diagnoses and drug claims." *Medical Care*, vol. 43, no. 1, pp. 34–43, 2005.

Chapter 5

PREDICTING 30-DAY RISK AND COST OF "ALL-CAUSE" HOSPITAL READMISSIONS

Predicting 30-Day Risk and Cost of "All-Cause" Hospital Readmissions

Shanu Sushmita, Garima Khulbe, Aftab Hasan, Stacey Newman
Center for Data Science, Institute of Technology, University of Washington, Tacoma
sshanu@uw.edu, garima1@uw.edu, aftabh@uw.edu, newmsc@uw.edu

Padmashree Ravindra, Senjuti Basu Roy, Martine De Cock, Ankur Teredesai
Center for Data Science, Institute of Technology, University of Washington, Tacoma
padmashree.ravindra@gmail.com, senjutib@uw.edu, mdecock@uw.edu, ankurt@uw.edu

Abstract

The hospital readmission rate of patients within 30 days after discharge is broadly accepted as a healthcare quality measure and cost driver in the United States. The ability to estimate hospitalization costs alongside 30 day risk-stratification for such readmissions provides additional benefit for *accountable care*, now a global issue and foundation for the U.S. government mandate under the Affordable Care Act. Recent data mining efforts either predict healthcare costs or risk of hospital readmission, but not both. In this paper we present a dual predictive modeling effort that utilizes healthcare data to predict the risk and cost of any hospital readmission ("all-cause"). For this purpose, we explore machine learning algorithms to do accurate predictions of healthcare costs and risk of 30-day readmission. Results on risk prediction for "all-cause" readmission compared to the standardized readmission tool (LACE) are promising, and the proposed techniques for cost prediction consistently outperform baseline models and demonstrate substantially lower mean absolute error (MAE).

1 Introduction

Patients with chronic conditions repeatedly get admitted to a hospital for treatment and care. They are often discharged when their condition stabilizes only to get readmitted again, many times within just a few days. This process is termed as *hospital readmissions*. The readmission problem in the U.S. is severe: currently one in five (20%) Medicare patients are readmitted to a hospital within 30 days of discharge. Three quarters of these readmissions (75%) are actually considered avoidable (Jencks, Williams, and Coleman 2009). In addition to raising red flags about gaps in quality of care, hospital readmissions also place a huge financial burden on the health system. In 2011, there were approximately 3.3 million adult 30-day all-cause hospital readmissions in the United States, and they were associated with about $41.3 billion in hospital costs (Hines et al. 2011). Avoidable readmissions account for around $17 billion a year (Jencks, Williams, and Coleman 2009). In the U.S., the readmission rate of patients at a hospital is tracked as a proxy for measuring the overall quality of treatment a patient has received, and, under the Affordable Care Act, Medicare has started penalizing hospitals that have higher-than-expected rates of 30-day readmissions[1].

In this paper we tackle two related problems, namely *(1) predicting whether a patient is at risk of being readmitted to the hospital within 30 days after discharge*, and *(2) estimating the cost of that hospital readmission*. The ability to prioritize a care plan along both of these variables can enable hospital systems to more effectively allocate the limited human and budgetary resources available to the high-risk individuals (i.e., higher-cost, earlier readmissions). Potential care transition gaps and targeted interventions can be derived from such models with a more profound impact on overall population management.

Existing dedicated efforts for accurately predicting 30-day risk of readmission are mostly focused on a specific cohort[2], such as congestive heart failure patients (Balla, Malnick, and Schattner 2008), cancer patients (Francis et al. 2015), emergency readmissions (Shadmi et al. 2015), etc. While these models are very useful, there is a lot of value in having *all-cause* risk and cost of readmission models that are not tied to a specific disease. In addition to allowing to derive risk and cost scores for patients who do not belong to any of the well studied cohorts, these models can also be used for incoming patients for which we do not know (yet) which cohort they belong to. To the best of our knowledge, none of the recent efforts predict cost or risk for all-cause readmissions, which is a completely different medical and data mining problem involving large, heterogeneous patient population sizes compared to disease specific cohorts such as heart failure.

In this study, we evaluate state-of-the-art machine learning techniques for predicting 30-day risk and cost on admission data of patients provided by a large hospital chain in the Northwestern U.S. We treat the risk prediction problem as a binary classification task, namely predicting whether the next admission of a given patient will be within 30 days or not. The LACE index is often used in clinical practice for this purpose (Zheng et al. 2015). This index considers

[1] http://www.cms.gov/Medicare/Medicare-Fee-for-Service-Payment/AcuteInpatientPPS/Readmissions-Reduction-Program.html, accessed on Oct 22, 2015

[2] A sub-group of a given population with similar characteristics (e.g., medical conditions), such as s group of diabetes patients.

four numerical variables, namely length of stay (L), acuity level of admission (A), comorbidity condition (C), and use of emergency rooms (E). The LACE score of a patient is obtained by summing up the values of these four variables at the time of discharge. A threshold (usually ≥ 10) is then set to determine which patients are at "high" readmission risk (Zheng et al. 2015). We use LACE as a baseline to compare the performance of the machine learning algorithms we investigate in this paper. We find that the use of machine learning techniques allows to achieve higher sensitivity (recall) without penalizing the specificity and precision too much. On the cost prediction side, we find that the simple baseline strategy of forecasting that the next admission of a patient will cost as much as the average of his previous admissions works reasonably well. In addition, a substantially lower mean absolute error (MAE) can be achieved with M5 model trees.

The rest of the paper is organized as follows: after giving an overview of related work in Section 2, we formalize the risk and cost prediction problems in Section 3. The machine learning algorithms applied in this paper for risk and cost predictions are explained in Section 4. The dataset and features are described in Section 5. In Section 6 we discuss the performance of the algorithms. Finally, in Section 7 we conclude with our overall findings.

2 Related Work

In this section, we give a brief overview of research efforts done independently along each of the two dimensions: readmission risk prediction and healthcare cost prediction. To the best of our knowledge, there is no existing work that studies risk and cost prediction problems in a combined way.

Healthcare Cost Prediction

Previously proposed cost prediction models often used rule-based methods and linear regression models. A challenge with the rule-based methods (e.g. (Kronick et al. 2002)) is that they require substantial domain knowledge which is not easily available and is often expensive. Linear regression models on the other hand are challenged by the skewed nature of healthcare data. Healthcare cost data typically features a spike at zero, and a strongly skewed distribution with a heavy right-hand tail (Jones 2010). As a result, the prediction models are posed with the challenge of an extreme value situation. This phenomenon is also observed in the dataset used in this study (see Figure 1). Consequently, several advanced statistical methods (in-sample estimation) have been proposed to overcome the skewness issue, such as General Linear Models (GLM) (Manning, Basu, and Mullahy 2005), mixture models (Mullahy 1997), etc. For a comprehensive comparison of previously proposed statistical methods for healthcare cost prediction, we refer to the review paper (Mihaylova et al. 2011). The development of healthcare cost prediction models using machine learning methods has been more recent (e.g., (Lahiri and Agarwal 2014; Sushmita et al. 2015)). (Lahiri and Agarwal 2014) investigate classification algorithms to predict whether an individual is going to incur higher or lower healthcare expenditure. (Sushmita et al. 2015) use three machine learning algorithms for cost prediction – regression tree, M5 model tree and random forest, and observe improved performance when compared to traditional methods. In this paper, we also investigate these algorithms for the task of predicting cost of hospital readmission. To the best of our knowledge, their utility for predicting the costs of hospital readmissions specifically (as opposed to predicting general healthcare costs) has not been investigated before.

Hospital Readmission Prediction

In 2011, there were approximately 3.3 million adult 30-day all-cause hospital readmissions in the United States, and they were associated with about $41.3 billion in hospital costs (Hines et al. 2011). Many of these hospitalizations are readmissions of the same patient within a short period of time. These readmissions act as a substantial contributor to rising healthcare costs (Jencks, Williams, and Coleman 2009). Readmission rates are also used as a screening tool for monitoring the quality of service and efficiency of care provided by healthcare providers (Balla, Malnick, and Schattner 2008). While predicting risk-of-readmission has been identified as one of the key problems for the healthcare domain, not many solutions are known to be effective (Krumholz et al. 2007; Ottenbacher et al. 2001). In fact, to improve the clinical process of heart failure patients for instance, healthcare organizations still leverage the proven best-practices, called "*Get With The Guidelines*" by the American Heart Association. In general, related work on risk-of-readmission prediction has primarily attempted to study cohort specific readmission risk, such as, heart failure, pneumonia, stroke, and asthma, but the effort of designing large scale machine learning algorithms for all-cause readmission is still at a rather rudimentary stage.

Despite several years of continued research efforts in modeling risk of readmission and healthcare cost, a dual predictive tool that utilizes healthcare data to predict risk and cost of hospital readmission has not been explored before. This study makes the first step in that research direction.

3 Problem Description

The goal of this study is to predict a patient's 30-day **risk** of hospital readmission and the associated **cost** of that readmission. We assume that the learning task at hand is a combination of a supervised classification problem (risk prediction) and a regression problem for predicting the cost (in dollars) of the readmission. The feature vector $X_i = (x_{i1}, x_{i2}, ..., x_{iM})$ of an instance i includes information about general demographics such as age and gender of the patient, as well as specific clinical and cost information at the time of discharge from the hospital. The goal is to produce an output vector $Y_i = (y_{i1}, y_{i2})$ consisting of a label y_{i1} that indicates whether the next admission of the patient will be in 30 days ("yes") or not ("no"), and the cost y_{i2} of the next admission. Let us use $\mathcal{X}$ to denote the set of all instances (feature vectors), and let $\mathcal{Y} = \{\text{yes, no}\} \times \mathbb{R}^+$ be the set of all dual labels. Given training examples of the form (X_i, Y_i) with $X_i \in \mathcal{X}$ and $Y_i \in \mathcal{Y}$, the aim is to

X (Input Vector)		Y (Output Vector)	
Admission		Next Admission	
Unique Identifier	Features (X)	Risk (y_1)	Cost (y_2)
id_1	$x_{11}, x_{12}, ...x_{1M}$	yes	\$45,132
id_1	$x_{21}, x_{22}, ...x_{2M}$	yes	\$41,305
id_1	$x_{31}, x_{32}, ...x_{3M}$	no	\$17,809
id_2	$x_{41}, x_{42}, ...x_{4M}$	yes	\$21,305
id_2	$x_{51}, x_{52}, ...x_{5M}$	no	\$55,809
id_3 (**test case**)	$x_{61}, x_{62}, ...x_{6M}$	?	?

Table 1: Example input and output scenario for the risk and cost prediction task. Here, the first column indicates a unique identifier for each patient in the dataset.

learn a model $\mathcal{H} : \mathcal{X} \rightarrow \mathcal{Y}$ that can label new, unseen instances from $\mathcal{X}$ with a dual label from $\mathcal{Y}$ in an accurate way. We address this multi-label prediction learning problem in a manner similar to binary relevance (Tsoumakas and Katakis 2007), by learning a model for each label:

- **Risk of 30-Day Readmission – Classification Task:** For given training examples of the form (X_i, y_{i1}), where $X_i \in \mathcal{X}$ and $y_{i1} \in \{\text{yes, no}\}$, the goal is to learn a model $\mathcal{H}_1 : \mathcal{X} \rightarrow \{\text{yes, no}\}$ that accurately predicts whether the next admission of a patient will be within 30 days.
- **Cost of Readmission – Regression Task:** For given training examples of the form (X_i, y_{i2}), where $X_i \in \mathcal{X}$ and $y_{i2} \in \mathbb{R}^+$ (cost in dollars), the goal is to learn a model $\mathcal{H}_2 : \mathcal{X} \rightarrow \mathbb{R}^+$ that accurately predicts the cost of the next admission.

The combined model is then obtained as $\mathcal{H}(X) = (\mathcal{H}_1(X), \mathcal{H}_2(X))$, for X in $\mathcal{X}$.

We also tried other techniques for multi-label prediction like – Label Powerset (Cherman, Monard, and Metz 2011) and Chain Classifier (Read et al. 2009), but initial evaluation results with these methods were not as good as those obtained with the binary relevance approach, so we omit them from this paper.

Table 1 illustrates the input and output representations for the risk and cost prediction problem. Let us assume that the patient with id_1 was admitted to the hospital four times (say on Jan 14, Feb 2, Feb 28 and Apr 15). That means that he had three *re*admissions, two within 30 days (high risk) with cost being \$45,132 and \$41,305 respectively, and one after 30 days (low risk) with cost being \$17,809. These response features are constructed based on attributes from the original, raw data (shown in Table 2). During the training phase, data of patient id_1 and id_2 will be used to train binary classifiers (to predict risk) and regression models (to predict cost); during the test phase the models will be used to predict the risk and the cost of patient id_3's next encounter.

We evaluate the accuracy of the learned models in several ways. Accuracy is traditionally measured as the percentage of instances that are classified correctly. It has been emphasized that the use of accuracy as an evaluation measure for data where there is an imbalance between positive and negative classes can yield misleading conclusions (Fatourechi et al. 2008; He and Garcia 2009). Readmission data is typically imbalanced. As Table 2 shows, approximately 27% of the admissions in our study are within 30 days (i.e. 27% positive instances), while the remaining 73% happen after 30 days (i.e., 73% negative instances). In addition to accuracy, we therefore also evaluate the binary classification models in terms of sensitivity (recall), specificity (true negative rate), and precision. Recall that sensitivity is TP/(TP + FN), specificity is TN/(TN+FP), and precision is TP/(TP+FP), with TP, FP, TN, and FN respectively denoting the number of true positives, false positives, true negatives and false negatives. The performance of the cost prediction algorithms is evaluated using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), with a lower error indicating a better performance.

4 Methods

In this section we give an overview of the machine learning algorithms used in this study. For risk-of-readmission prediction we used state-of-the-art classification techniques, while for cost prediction we used regression techniques.

Risk Prediction Methods

Support Vector Machine: A Support Vector Machine (SVM) is a statistical learning method for training classifiers based on different types of kernel functions – polynomial functions, radial basis functions, etc. An SVM learns a linear separating hyperplane by maximizing the margin between the classes (Drucker et al. 1996). The decision boundary is maximised with respect to the data points from each class (known as support vectors) that are closest to the decision boundary. For this study, we tested SVM with linear and radial kernel, and we report the results for radial in Table 3 because its performance was better than the linear kernel settings. We also tested for different regularization parameters ($C = 1, 5, 10, 15$), but the overall results did not change.

Logistic Regression: Logistic Regression is an example of a discriminative classifier that models the posterior $p(y_1|X)$ directly given the input features. That is, it learns to map the input (X) vector directly to the output class label y_1 (risk in our case). When the response is a binary (dichotomous) variable, logistic regression fits a logistic curve to the relationship between X and y_1 (Ng and Jordan 2001). The class decision for the given probability is then made based on a threshold value. The threshold is often set to 0.5, i.e. if $p(y_1|X) \geq 0.5$, then we predict that the next readmission of the patient will be within 30 days, and otherwise not. We tested with multiple threshold values to make the class decision.

Decision Trees: An alternative approach to linear classification is to partition the space into smaller regions, where the interactions are more manageable. Like for the other methods described in this section, the goal of a classification tree is to predict a response y_1 (risk in our case) from inputs X. This is done by growing a binary tree. At each internal node in the tree, a test is applied to one of the inputs, and depending on the outcome, the left or the right subbranch of the tree is then selected. Eventually a leaf node is reached, where the prediction is made. For this study, we used an implementation of the classification and regression tree algorithm (CART) (Breiman et al. 1984) in R. We tested the performance of classification trees using different complexity parameters (cp = 0.01, 0.001, 0.0005). In Table 3 we

report the results of the best performing tree with cp set to 0.01 value.

Random Forest: Random forest regression is an ensemble learning method that operates by constructing a multitude of regression trees at training time and outputting the mean prediction of the individual trees for new observations. Each tree is constructed using a random sample of the observation and feature space from the original dataset. This has the effect of correcting the tendency of individual regression trees to overfit the training data (Breiman 2001).

Generalised Boosted Modeling (GBM): Boosting is an approach to machine learning based on the idea of creating a highly accurate predictive model ensemble by combining many relatively weak and inaccurate models (Freund and Schapire 1997). In other words, boosting is an optimization technique that minimizes the loss function by adding, at each step, a new model that best reduces the loss function. It is often used to grow an ensemble of classification trees, like we do in this paper. In this study we use the gbm implementation of AdaBoost in R, which is an implementation of extensions to Freund and Schapire's AdaBoost algorithm and Friedman's gradient boosting machine[3].

All the models are trained and tested using R[4]. Additionally, we also set the output of each model to be the class probability (prob= TRUE), instead of the class labels. This was done in order to test for different decision threshold values $(0.0 - 1.0)$ to find the optimal balance between different evaluation measures. We report results for thresholds between $0.2 - 0.52$ in Figures 2-5.

Cost Prediction Methods

Linear Regression: Linear regression is used extensively in the literature on healthcare cost prediction, so, even though it has its limitations, it can not be ignored in this study. We use a linear regression model to predict cost using an M-dimensional vector of predictive variables (see Table 2).

M5 Model Tree: M5 model trees are a generalization of the CART model (Breiman et al. 1984). The structure of an M5 model tree follows that of a decision tree, but has multiple linear regression models at the leaf nodes, making the model a combination of piecewise linear functions. The algorithm for the training of a model tree breaks the input space of the training data through a recursive partitioning process similar to the one used in CART. After partitioning, linear regression models can be fit on the leaf nodes, making the resulting regression model locally accurate.

In addition to the linear regression and M5 Model Tree methods, **decision trees** and **generalised boosted modeling (GBM)** as described for risk prediction task were also used for predicting the cost.

5 Dataset and Features

The study in this paper includes admission data of patients provided by a large hospital chain in the Northwestern United States. Each admission record includes demographic information (e.g., gender, ethnic group), clinical information (e.g., primary diagnosis), care provider details, administrative data (e.g., length of stay) and billing information (e.g., charge amount). First, we performed data filtering as part of data pre-processing. Of the available ∼221K admission records, we excluded instances of admissions for which the patient died before discharge, or was transferred to another acute care facility within the hospital chain, or left against medical advice. Additionally, we excluded records where the next admission date is unknown, since they cannot be used to evaluate the correctness of cost and risk of readmission prediction. We also excluded hospitalizations with unspecified primary diagnosis.

Next, we performed several feature engineering steps. There were 214 features in the raw data. We used a forward stepwise regression approach (Derksen and Keselman 1992) to select a subset of this feature set. This subset is shown in Table 2. Most of the features from Table 2 correspond directly to features from the raw data; others have been constructed based on previous history of the patient. That is, most of the features are drawn from individual admission records, but some are aggregated across multiple admission records of the same patient. The features from the latter category are:

- **Number of Comorbidities:** this is the total number of unique comorbidities[5] that were registered for a patient up to the time of discharge. We used the Elixhauser comorbidity (Elixhauser et al. 1998) information of a patient to identify all comorbidities associated to that patient. Comorbidity is associated with worse health outcomes, increased healthcare costs and is known to impact prediction of risk of readmission (Donze et al. 2013). Therefore, we use it as one of the predictor variables.

- **Number of Existing conditions:** this is is the total number of unique diagnoses registered for this patient up to this point, including during previous admissions. The list of existing conditions of a patient is represented using ICD9-CM codes in the raw data (∼4K distinct values). We grouped the ICD9-CM codes using Clinical Classification Software (CCS)[6], and included the count of distinct CCS codes per patient as a feature.

- **Number of Previous Admissions:** this is the total number of hospital admissions registered for this patient up to this point. Here, the assumption is that a patient with a history of several hospital admissions is more likely to be readmitted again.

In this paper we use frequency counts (e.g. Number of Comorbidities) to overcome the limitation of significant sparseness in this dataset, for future research, we aim to explore statistical methods to overcome this limitation. Finally, we randomly sampled ∼10K instances with the feature set shown in Table 2 to train and test our models.

[3] http://cran.r-project.org/web/packages/gbm/gbm.pdf

[4] http://www.r-project.org

[5] Two or more coexisting medical conditions or disease processes that are additional to an initial diagnosis.

[6] https://www.hcup-us.ahrq.gov/toolssoftware/ccs/ccs.jsp

Feature	Type	Distribution
Gender	Categorical	Female (5,818) Male (4,176)
Adult	Boolean	Yes (9,792) No (202)
Age $\geq$ 65	Boolean	Yes (4,801) No (5,193)
Ethnic Group	Categorical	Caucasian (8,303) African American (669) Hispanic/Latino (257) American Indian (185) Asian (172) Pacific Islander (157) Multi-Racial (83) Non-Hispanic (25) Middle Eastern (18) Eskimo (4) Other (121)
Marital Status	Categorical	Single (2,387) Married (4,148) Widowed (1,939) Divorced (1,084) Separated (211) Significant Other (192) Legally Separated (23) Domestic Partner (4) Other (2) Unknown (4)
Admit Type	Categorical	Emergency (7,350) Elective (2,519) Urgent (86) Trauma Center (39)
Financial Class	Categorical	Medicare (5,595) Medicaid (924) Self-pay (319) Other (3,156)
Care Type	Categorical	Acute (9,975) Geropsychiatric (19)
No. of Comorbidities	Numeric	Mean: 6.43 (SD = 4.25)
No. of Existing Conditions	Numeric	Mean: 10.67 (SD = 8.05)
Length of Stay (Days)	Numeric	Mean: 4.28 (SD = 4.41)
Same Day Discharge	Boolean	Yes (105) No (9,889)
Blood Pressure at Discharge	Categorical	80-89 or 120-139 (4,189) $<$80 and $<$ 120 (3,529) 90-99 or 140-159 (1,557) $>$ 99 or $>$ 159 (719)
No. of Previous Admissions	Numeric	Mean: 1.55 (SD = 2.87)
Next Admission $<$ 30 Days **(Response Variable)**	Categorical	Yes (2,697), 27% No (7,297), 73%
Additional Features for Cost Prediction		
Current Admit Cost ($)	Numeric	Mean: 53,530 (SD = 72,888)
Current Bed Charge ($)	Numeric	Mean: 10,120 (SD = 14,458)
Cost of Next Readmission ($) **(Response Variable)**	Numeric	Mean: 54,140 (SD = 74,400)

Table 2: Overview of the feature set used in the prediction of "risk" and "cost" of readmission

6 Result Analysis

We evaluated the machine learning methods from Section 4 for the risk and cost of hospital readmission prediction problems described in Section 3 on the dataset described in Section 5. In this section we present the results and discuss the key observations.

Risk Prediction

The results of the five machine learning algorithms as well as the LACE baseline, are presented in Figures 2-5 and Table 3. Among the existing risk prediction tools, the LACE index is regularly used in hospitals (Zheng et al. 2015). This index considers four numerical variables, namely length of stay (L), acuity level of admission (A), comorbidity condition (C), and use of emergency rooms (E). A LACE score is obtained by summing up the values of these four variables. A threshold (usually $\geq$ 10) is then set to determine patients with "high" readmission risk (Zheng et al. 2015). We use

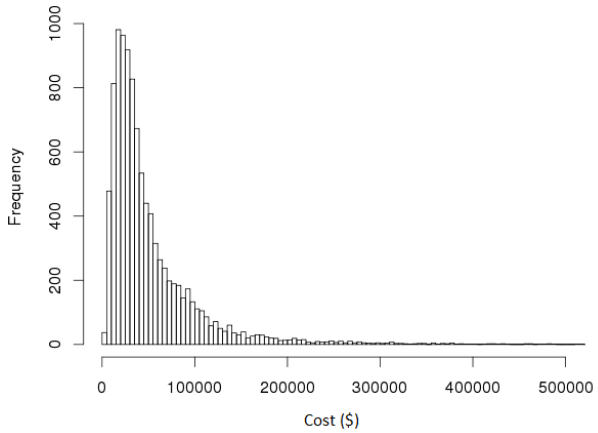

Figure 1: Distribution of hospital readmission cost in the readmission dataset

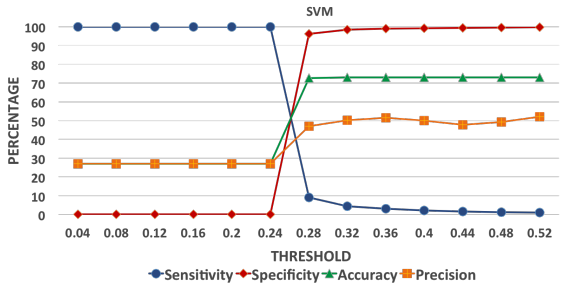

Figure 2: Risk prediction performance results of SVM

LACE as a baseline to compare the performance of the machine learning algorithms we investigate in this paper.

We evaluated the models developed with all five machine learning methods using 10-fold cross-validation across different threshold values (see Figures 2-5). This was done so that a threshold value that would give the highest possible sensitivity, but at the same time also have comparable specificity to that of the LACE tool can be identified. It should be noted that for the 30-day risk of readmission prediction task, higher sensitivity is more desirable. This is because correctly identifying the "high risk" patients who are likely to be readmitted within 30 days is more crucial than correctly identifying low risk patients (discussed in Section 1). Overall results corresponding to the best threshold values for all the models are shown in Table 3, and Figure 6 shows the trade-off between sensitivity and specificity for all the models.

There are three key observations to be made from these results. First, the results in Table 3 suggest that most machine learning methods show promising results when compared to the baseline LACE method. Not only was it possible to achieve higher sensitivity than LACE, but this was done without penalizing the specificity and precision too much. More in detail, with 3 out of the 5 machine learning methods we achieved a sensitivity of over 80%, while the results for specificity and precision remained comparable to that of

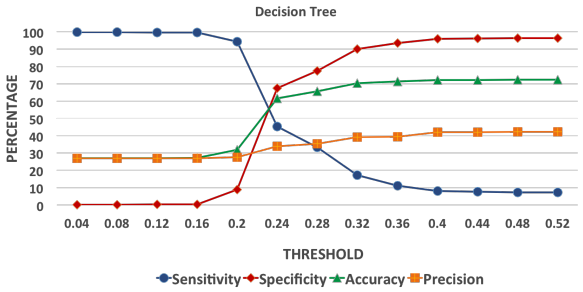

Figure 3: Risk prediction performance results of Decision Trees

Algorithm	Sensitivity (%)	Specificity (%)	Precision (%)
LACE	76.42	38.95	31.63
SVM	98.11	1.84	26.98
Decision Trees	94.07	9.04	27.65
Random Forest	84.76	25.60	29.63
Logistic Regression	92.47	13.24	28.26
GBM	90.43	18.24	29.02

Table 3: Performance comparison of different machine learning methods for the task of predicting whether the next hospital admission of a patient will be within 30 days. The results are based on 10-fold cross-validation.

LACE (sensitivity = 76%). The sensitivity results for the other two methods, namely SVM and decision tree, were also very high (sensitivity $\geq$ 94%) and the precision score was comparable, but the proportion of "low risk" instances which were correctly identified was very low (specificity $\leq$ 9%).

Second, the rate of change in sensitivity and specificity slightly differs across different machine learning methods. For instance, as the threshold values increase, the sensitivity and specificity in the decision trees, logistic regression, and generalised boosted models exhibit sigmoid curves (see Figure 3, and 5), characterized by a small progression in the beginning and then accelerating and converging over larger threshold values. For random forest models, the change is almost linear (Figure 4). For SVM a steep drop in sensitivity and rise in specificity is observed between 0.24 to 0.28 threshold values (Figure 2). Further investigation of the

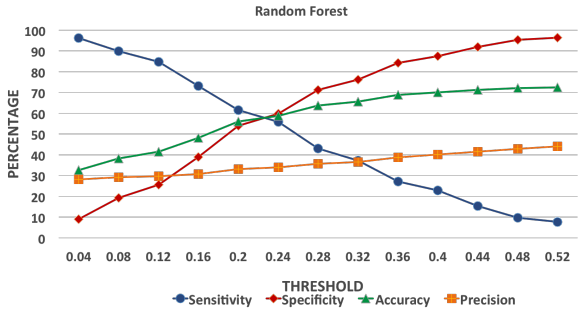

Figure 4: Risk prediction performance results of Random Forest

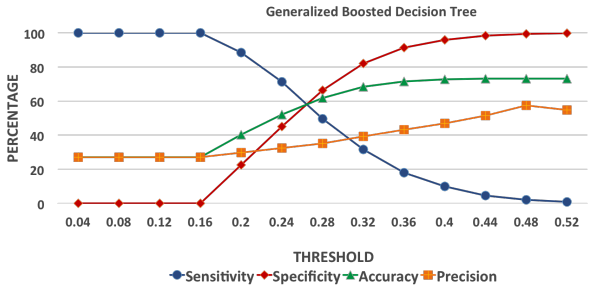

Figure 5: Risk prediction performance results of GBM

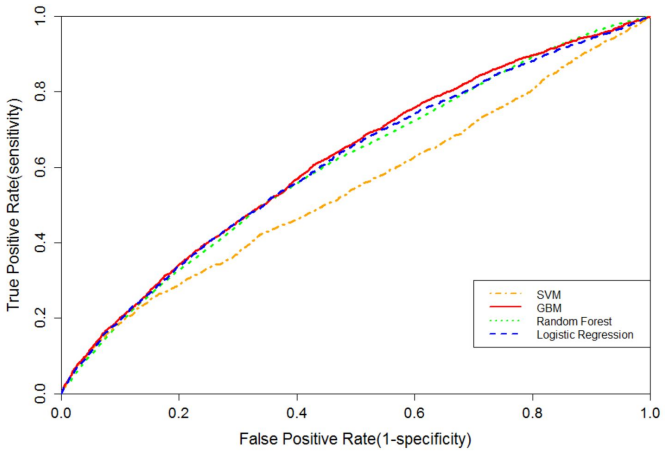

Figure 6: ROC curve comparing risk prediction performance results. It shows the trade-off between sensitivity and specificity. It can be seen that GBM is the best classifer, while SVM is the worst.

SVM results showed that there was big increase in the number of true negatives and false negatives around these threshold values, illustrating that SVM is less robust than the other methods, and that its good performance depends on fine tuning of the cutoff threshold.

The third key observation is that, across all methods, 50-60% is the maximum score that can be achieved when a "perfect" balance across all measures (sensitivity, specificity, accuracy and precision) is desired. This is a meaningful result because it shows that the machine learning methods can give good performance for the majority ($>$ 50%) of instances across all measures. It is interesting to observe in Figures 2-5 that this optimal point of balance emerges within the same small range of threshold values across all different machine learning methods.

Overall, for the risk prediction task, the results for most machine learning methods for any type of readmission ("all-cause") are promising when compared to a standardized risk prediction tool (LACE). It was possible to achieve higher sensitivity (recall) without penalizing the specificity and precision too much. Improving the precision and specificity further will be a task to explore in future.

Cost Prediction

We measured the performance of the methods for cost prediction using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). A lower error indicates that the predicted dollar amount is closer to the actual cost. As for risk prediction, we evaluated all models using 10-fold cross-validation. An overview of MAE and RMSE results is presented in Table 4. We compared the results of four machine learning methods, namely linear regression, M5 model tree, generalised boosted model and decision tree, with those of two baseline methods:

- **Average Baseline (AB):** the Average Baseline (AB) measure is the overall mean cost μ of individual average encounter costs for all the beneficiaries within the training set prior to the current encounter for which we are predicting the cost. This mean (μ) score is then used as the baseline predicted cost for all patient-encounter pairs in the test set.
- **Current Admit Cost (CAC):** the Current Admit Cost (CAC) baseline model is a linear regression model fitted using only the current admission cost during the training period as a predictor variable, with next readmission cost being the response variable. Note that the difference between this current admit cost baseline and the competing linear regression model is that all features (as shown Table 2) from the readmission dataset were used to train the linear regression model, while only the 'current admit cost' variable was used in the CAC baseline model.

Algorithm	MAE ($)	RMSE ($)
Average Baseline (AB)	21,609	27,176
Current Admit Cost (CAC)	20,882	26,458
Linear Regression (LM)	20,232	26,124
M5 Model Tree (M5)	18,263	24,824
Generalised Boosted Model (GBM)	20,065	26,388
Decision Tree (DT) (cp=0.01)	20,512	26,328

Table 4: Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) in dollars for the cost prediction task

As can be seen in Table 4, for all-cause readmissions, our data mining models exhibit lower prediction error compared to the Average Baseline (AB) method in terms of both MAE and RMSE. Within that, M5 model tree has the lowest prediction error. The errors for the Current Admit Cost (CAC) baseline method were interestingly enough comparable to the errors of several of the more sophisticated methods.

Overall, two key observations can be made from the performance results shown in Table 4. First, current admit cost and average cost are strong baseline models, and therefore current hospital admission cost or average cost alone can be a good indicator for the next readmission cost provided it is available to the care provider. Second, among all machine learning algorithms, M5 model tree performed best and achieved a substantially lower MAE than strong baseline models for predicting next readmission cost.

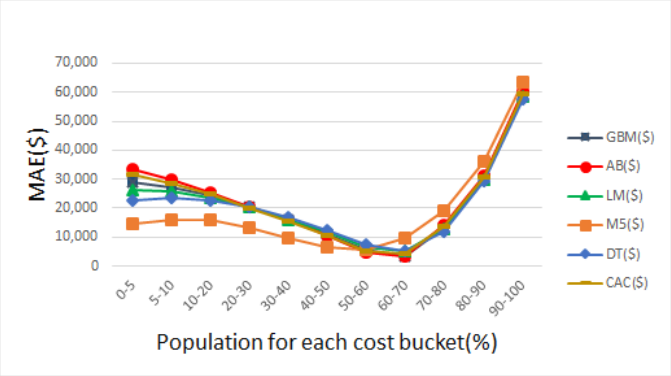

Figure 7: Comparison of average Mean Absolute Error (MAE) across different cost buckets.

The knowledge that the cost distribution in our dataset is highly skewed (see Figure 1), as is known to be the case with healthcare costs in general, inspired us to delve deeper into investigating for which fraction of the patients our models can predict costs with error margins that are reasonably bounded. To this end, we divided the population into 11 different cost buckets, shown on the horizontal axis in Figure 7. The cost buckets range from the 5% lowest cost patients (subpopulation 0-5%) to the 10% highest cost patients (subpopulation 90-100%). Next, for each of our cost prediction methods, we measured the average MAE over all patients within each subpopulation. The results are shown in Figure 7.

It is interesting to observe that all methods display a similar behavior: the predictions across all models are most accurate within the middle of the range, i.e. for moderate cost patients (40-70%). For the low cost patients (0-40%), the machine learning techniques clearly outperform the baseline models. This is especially the case for the M5 model tree. For the high-cost patients (70-100%) it is interestingly enough the other way around, although the difference in error between the different techniques is relatively small compared to the size of the actual healthcare costs in this case. Still, the results in Figure 7 indicate that it might be beneficial to train a hierarchical model that first predicts a cost bucket and then uses a model trained specifically for that cost bucket to arrive at a final prediction in dollars.

7 Conclusion

The rate of hospital readmissions of patients is a key measure that is tracked for numerous reasons. Consequently, risk stratification of a population and readmission models are becoming increasingly popular. Recent data mining efforts either predict healthcare costs or risk of hospital readmission, but not both. The goal of this study was a dual predictive modeling effort that utilizes healthcare data to predict the risk and cost of any hospital readmission ("all-cause"). For this purpose, we explored machine learning algorithms to do accurate predictions for risk and cost of 30-day readmission. For the task of risk prediction, results for most machine learning methods for any type of readmission ("all-cause")

were promising when compared to a standardized risk prediction tool (LACE). It was possible to achieve higher sensitivity (recall) without penalizing the specificity and precision too much. On the cost prediction side, two key observations were made from the performance results of the machine learning methods. First, average admission cost and current admission cost are strong predictors, and therefore they alone can be a good indicator for the next readmission cost. Second, among the four machine learning algorithms, M5 model tree consistently performed better and achieved a substantially lower MAE than strong baseline models for predicting next readmission cost.

Acknowledgments

The authors would like to thank MultiCare[7] for providing access to the readmission dataset used in this study. This research was made possible thanks to generous support of Edifecs Inc. and a Microsoft Azure for Research grant.

References

Balla, U.; Malnick, S.; and Schattner, A. 2008. Early readmissions to the department of medicine as a screening tool for monitoring quality of care problems. *Medicine* 87(5):294–300.

Breiman, L.; Friedman, J. H.; Olshen, R. A.; and Stone, C. J. 1984. *Classification and Regression Trees*. Wadsworth Publishing Company.

Breiman, L. 2001. Random forests. *Machine Learning* 45(1):5–32.

Cherman, E. A.; Monard, M. C.; and Metz, J. 2011. Multi-label problem transformation methods: a case study. *CLEI Electron. J.* 14(1).

Derksen, S., and Keselman, H. J. 1992. Backward, forward and stepwise automated subset selection algorithms: Frequency of obtaining authentic and noise variables. *British Journal of Mathematical and Statistical Psychology* 45(2):265–282.

Donze, J.; Aujesky, D.; Williams, D.; and Schnipper, J. L. 2013. Potentially avoidable 30-day hospital readmissions in medical patients: derivation and validation of a prediction model. *JAMA Internal Medicine* 173(8):632–638.

Drucker, H.; Burges, C. J. C.; Kaufman, L.; Smola, A. J.; and Vapnik, V. 1996. Support vector regression machines. In *Advances in Neural Information Processing Systems 9, Proceedings of the 1996 NIPS conference*, 155–161.

Elixhauser, A.; Steiner, C.; Harris, D. R.; and Coffey, R. M. 1998. Comorbidity measures for use with administrative data. *Medical care* 36(1):8–27.

Fatourechi, M.; Ward, R. K.; Mason, S. G.; Huggins, J.; Schlögl, A.; and Birch, G. E. 2008. Comparison of evaluation metrics in classification applications with imbalanced datasets. In *7th International Conference on Machine Learning and Applications (ICMLA)*, 777–782.

Francis, N. K.; Mason, J.; Salib, E.; Allanby, L.; Messenger, D.; Allison, A. S.; Smart, N. J.; and Ockrim, J. B. 2015. Factors predicting 30-day readmission after laparoscopic colorectal cancer surgery within an enhanced recovery programme. *Colorectal Disease* 17(7):148–154.

Freund, Y., and Schapire, R. E. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer System Science* 55(1):119–139.

He, H., and Garcia, E. A. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering* 21(9):1263–1284.

Hines, A. L.; Barrett, M. L.; Jiang, J.; and Steiner, C. A. 2011. Conditions with the largest number of adult hospital readmissions by payer. Technical report, Healthcare Cost and Utilization Project (HCUP).

Jencks, S. F.; Williams, M. V.; and Coleman, E. A. 2009. Rehospitalizations among patients in the Medicare fee-for-service program. *New England Journal of Medicine* 360(14):1418–1428.

Jones, A. 2010. Models For Health Care. Technical report, HEDG, c/o Department of Economics, University of York.

Kronick, R.; Gilmer, T. P.; Dreyfus, T.; and Ganiats, T. G. 2002. CDPS-Medicare: The chronic illness and disability payment system modified to predict expenditures for Medicare beneficiaries. Technical report.

Krumholz, H. M.; Normand, S. L. T.; Keenan, P. S.; Lin, Z. Q.; Drye, E. E.; Bhat, K. R.; Wang, Y. F.; Ross, J. S.; Schuur, J. D.; and Stauffer, B. D. 2007. *Hospital 30-day heart failure readmission measure methodology. Report prepared for the Centers for Medicare & Medicaid Services*.

Lahiri, C. B., and Agarwal, N. 2014. Predicting healthcare expenditure increase for an individual from Medicare data. In *Proceedings of the ACM SIGKDD Workshop on Health Informatics*.

Manning, W. G.; Basu, A.; and Mullahy, J. 2005. Generalized modeling approaches to risk adjustment of skewed outcomes data. *Journal of Health Economics* 24(3):465–488.

Mihaylova, B.; Briggs, A.; O'Hagan, A.; and Thompson, S. G. 2011. Review of statistical methods for analysing healthcare resources and costs. *Health Economics* 20(8):897–916.

Mullahy, J. 1997. Heterogeneity, excess zeros, and the structure of count data models. *Journal of Applied Econometrics* 12(3):337–50.

Ng, A. Y., and Jordan, M. I. 2001. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In *Advances in Neural Information Processing Systems 14, Proceedings of the 2001 NIPS conference*, 841–848. MIT Press.

Ottenbacher, K.; Smith, P.; Illig, S.; Linn, R.; Fiedler, R.; and Granger, C. 2001. Comparison of logistic regression and neural networks to predict rehospitalization in patients with stroke. *Journal of Clinical Epidemiology* 54(11):1159–1165.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2009. Classifier chains for multi-label classification. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, 254–269.

Shadmi, E.; Flaks-Manov, N.; Hoshen, M.; Goldman, O.; Bitterman, H.; and Balicer, R. D. 2015. Predicting 30-day readmissions with preadmission electronic health record data. *Med Care* 53(3):283–289.

Sushmita, S.; Newman, S.; Marquardt, J.; Ram, P.; Prasad, V.; De Cock, M.; and Teredesai, A. 2015. Population cost prediction on public healthcare datasets. In *Proceedings of ACM Digital Health 2015 (5th International Conference on Digital Health)*, 87–94.

Tsoumakas, G., and Katakis, I. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining* 3(1):1–13.

Zheng, B.; Zhang, J.; Yoon, S. W.; Lam, S. S.; Khasawneh, M.; and Poranki, S. 2015. Predictive modeling of hospital readmissions using metaheuristics and data mining. *Expert Systems with Applications* 42(20):7110–7120.

[7] http://www.multicare.org

Chapter 6

FAST, PRIVACY PRESERVING LINEAR REGRESSION OVER DISTRIBUTED DATASETS BASED ON PRE-DISTRIBUTED DATA

Fast, Privacy Preserving Linear Regression over Distributed Datasets based on Pre-Distributed Data

Martine De Cock
University of Washington
Tacoma
martine@uw.edu

Rafael Dowsley
Karlsruhe Institute of
Technology
rafael.dowsley@kit.edu

Anderson C. A.
Nascimento
University of Washington
Tacoma
andclay@uw.edu

Stacey C. Newman
University of Washington
Tacoma
newmsc8@uw.edu

ABSTRACT

This work proposes a protocol for performing linear regression over a dataset that is distributed over multiple parties. The parties will jointly compute a linear regression model without actually sharing their own private datasets. We provide security definitions, a protocol, and security proofs. Our solution is information-theoretically secure and is based on the assumption that a Trusted Initializer pre-distributes random, correlated data to the parties during a setup phase. The actual computation happens later on, during an online phase, and does not involve the trusted initializer. Our online protocol is orders of magnitude faster than previous solutions. In the case where a trusted initializer is not available, we propose a computationally secure two-party protocol based on additive homomorphic encryption that substitutes the trusted initializer. In this case, the online phase remains the same and the offline phase is computationally heavy. However, because the computations in the offline phase happen over random data, the overall problem is embarrassingly parallelizable, making it faster than existing solutions for processors with an appropriate number of cores.

Keywords

Secure Machine Learning, Private Linear Regression, Unconditional Security, Commodity Based Model

1 Introduction

Linear regression is a technique for modelling the relationship between one or more input variables – often called explanatory variables – and a real valued outcome. It is widely used as a tool for statistical analysis and is very popular as a technique to build predictive models in machine learning (see e.g. [30]). Linear regression models owe their popularity to various factors, including the fact that they can be trained efficiently, are intuitive, and fit the data reasonably well for many learning problems. In addition, the high inductive bias caused by the simplicity of the model helps prevent it from overfitting to a specific set of training examples. Furthermore, linear regression models do not require the outcome variable to be linear in terms of the input variables; they are simply linear in terms of the weights associated to each of the input variables.

Techniques for constructing a linear regression model, like other traditional machine learning (ML) methods, require that all training attributes and tuples be accessible to the learning algorithm. As storage and computing becomes increasingly distributed and heterogeneous, data movement and transformations become non-trivial, making conventional ML algorithms difficult to apply. Moreover, multiple parties often cannot or will not share their data due to economic incentives or privacy legislation, creating a dire need for privacy-preserving ML techniques.

The importance of such techniques is immediately apparent in ML applications in security-sensitive industries such as the financial sector and electronic surveillance. In the former, a bank may want to hire an analytics company to mine the data of its customers but, being bound by the customer agreement, cannot simply hand over the data to that company. In the latter, Internet service providers wanting to have a consulting firm do traffic analysis on their logs may be unwilling to disclose details about their customer base in the process. Another prominent example is the healthcare ecosystem. In addition, it is widely acknowledged that big data analytics can revolutionize the healthcare industry, among other things, by optimizing healthcare spending at all levels from patients to hospitals to governments. Important challenges for this reform are leveraging existing large and varied clinical and claims datasets to estimate future healthcare costs and taking measures in care-management that reduce such costs while improving overall population health. However, in practice a major roadblock is that ML for many healthcare tasks (e.g., estimating the risk of hospital readmission) needs data that is split over many different owners – healthcare providers, hospitals, and medical insur-

AISec'15, October 16, 2015, Denver, Colorado, USA.

DOI: http://dx.doi.org/10.1145/2808769.2808774.

ance companies – who do not want to or legally cannot share their data with outside entities.

In this paper, we attack the problem of securely computing a linear regression model between two parties that are not allowed to share their data. We propose protocols that securely compute a linear regression model over two separate datasets according to our definition (we later show that our solution can be extended to the case of multiple parties). Our results are information-theoretically secure and work in the commodity based model [3, 5]. This model can provide us with unconditionally secure protocols, that is protocols that do not rely on computational assumptions for ensuring their security. It has been proven that commitments [28, 7, 24], oblivious transfer [3, 2], distributed inner product [12], verifiable secret sharing [25, 13], and oblivious polynomial evaluation [31] are implementable in this setting. [20] presents recent general results on the power of the commodity based model. In the commodity based model, Alice and Bob have correlated data that was pre-distributed at the beginning of the protocol. The pre-distributed data can be seen as data provided by a trusted initializer during an offline setup phase. Note that, in this case, this trusted party does not engage in the protocol after the setup phase and never learns Alice and Bob's inputs.

If a trusted initializer is not available or desirable, we present a protocol where Alice and Bob can simulate the trusted initializer by running a two-party secure computation protocol by themselves during an offline setup phase. The remaining online phase remains the same. In this case, the overall protocol becomes computationally secure.

The online phase of our protocol is extremely efficient, having solely modular additions and multiplications. The offline, pre-processing phase, in the case of the computationally secure protocol, is based on any additive homomorphic public key cryptosystem. Because all the data used during the pre-processing is random, the computations to be performed become embarrassingly parallelizable and gains proportional to the number of available cores can be obtained, making even the costly pre-processing phase practical.

We improve the running time of previous algorithms for secure linear regression from days [19] to seconds in the online phase. Even when considering the time needed for Alice and Bob to perform their pre-processing during the offline phase (in the case of the computationally secure protocol) the overall computing time (offline and online phase) is in the order of minutes.

This paper is structured as follows: after discussing related work (Section 2) and giving preliminaries on the security model in Section 3, we present a high level overview of both our protocols for secure linear regression in Section 4. Next, we provide details on our secure computation of multiplication and inverse of matrices (Section 5 and 7), as well as how we deal with real numbers (Section 6). In Section 8, we summarize how these building blocks fit together in our information-theoretically secure protocol, while in Section 9 we explain how to substitute the trusted initializer and obtain a computationally secure protocol. In Section 10 we present runtime results of both protocols on ten different datasets, with a varying number of instances and features, showing a substantial speed-up compared to existing work. Finally, we sketch how to extend the protocols to more than two parties (Section 11) and how to obtain security against malicious adversaries (Section 12).

2 Related Work on Secure Linear Regression

There are many attempts in the literature at obtaining secure linear regression protocols over distributed databases. Most of them clearly do not even aim at obtaining the level of privacy usually required by modern cryptographic protocols (such as Karr et al. [21] and Du et al.[14], see also [29, 22]).

The pioneering work of Hall et al. [19] actually presents a protocol that achieves cryptographic security within the framework of secure two-party protocols and simulation based definitions of security, as proposed by Goldreich in [17]. However, we remark that as some of the protocols they propose rely on approximations, rather than exact computations, the correct framework to be used is that of Feigenbaum et al. [15, 16], instead of Goldreich's. Additionally, Hall et al. [19] uses operations in a finite field and homomorphic encryption as a building block. However, the (interesting version of the) linear regression problem deals with numbers in $\mathbb{R}$, or at least in $\mathbb{Q}$. To cope with this problem, a fixed-point data type and its representation in the finite field are defined in [19]. In such an approach, it is necessary to perform a truncation after each multiplication, but the description of the truncation protocol of [19] has a small (correctable) problem as explained in Appendix A. Finally, the overall computing time for solving the linear regression problem for 51K input vectors, each with 22 features, is two days [19]. The online phase of our protocol solves this problem in a few seconds. Even when considering the running time of the offline phase of our computationally secure protocol, by exploiting its embarrassingly parallelization property, the overall running time is still in the order of minutes for such a number of features and vectors.

In [26], a solution is proposed based on homomorphic encryption and garbled circuits for a scenario where many parties upload their data to a third party responsible for obtaining the regression model (with the help of a Crypto Service Provider, responsible for performing heavy cryptographic operations). The Crypto Service Provider is a semi-honest trusted party that is assumed to not collude with other players and actively engages in the protocol during its execution. In our information theoretical solution the trusted initializer does not engage in the protocol after the setup phase and our online phase is still much faster than the protocol presented in [26]. Even when we add up the offline phase and the online phase running times, in the case of our computationally secure protocol, when multiple cores are available for the offline phase computations, the overall running time is less for our protocol.

Finally, we assessed our secure linear regression by implementing and analyzing the results using ten real datasets. We chose a variety of different datasets based on their number of features and instances. Some of our datasets have millions of vectors. We are unaware of any other work on secure linear regression where real datasets of this size have been analysed before. For example, in [19] and in [26], the real datasets used had thousands of vectors.

3 Security Model

3.1 Secure Two-Party Computation

We consider honest-but-curious adversaries (i.e., adversaries that follow the protocol instructions but try to learn additional information about the other parties' inputs) and

define (exact) secure two-party computation following Goldreich [17].

A two-party computation is specified by an ideal functionality that is a (possibly randomized) mapping f: $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$ from inputs (a,b) to outputs (c,d). Let $f_1(a,b)$ denote the first output of f and $f_2(a,b)$ the second. A two-party protocol is a pair of polynomial-time interactive algorithms $\pi = (\pi_{\mathsf{Alice}}, \pi_{\mathsf{Bob}})$. Alice executes π_{Alice} with input a and randomness r_{Alice} and Bob executes π_{Bob} with input b and randomness r_{Bob}. The execution proceeds in rounds, each party is able to send one message in each round to the other party. The messages are specified by π, given the party's view, which consists of his input, randomness, and messages exchanged so far. Each party can also terminate, at any point, outputting some value based on his view. Let $\mathsf{view}^{\pi}_{\mathsf{Alice}}(a,b)$ denote Alice's view of the protocol execution, i.e, her input, her randomness, and all the exchanged messages. Let $\mathsf{view}^{\pi}_{\mathsf{Bob}}(a,b)$ similarly denote Bob's view of the protocol execution. Let $\mathsf{output}^{\pi}_{\mathsf{Alice}}(a,b)$ and $\mathsf{output}^{\pi}_{\mathsf{Bob}}(a,b)$ denote Alice's and Bob's outputs respectively.

DEFINITION 3.1. *A protocol π privately computes f with statistical security if for all possible inputs (a,b) the following properties hold:*

- *Correctness:*

$$\{\mathsf{output}^{\pi}_{\mathsf{Alice}}(a,b), \mathsf{output}^{\pi}_{\mathsf{Bob}}(a,b)\} \equiv \{f(a,b)\}$$

- *Privacy: There are simulators $\mathcal{S}_{\mathsf{Alice}}$ and $\mathcal{S}_{\mathsf{Bob}}$ such that:*

$$\{\mathcal{S}_{\mathsf{Alice}}(a,c), f_2(a,b)\} \stackrel{s}{\approx} \{\mathsf{view}^{\pi}_{\mathsf{Alice}}(a,b), \mathsf{output}^{\pi}_{\mathsf{Bob}}(a,b)\}$$

$$\{f_1(a,b), \mathcal{S}_{\mathsf{Bob}}(b,d)\} \stackrel{s}{\approx} \{\mathsf{output}^{\pi}_{\mathsf{Alice}}(a,b), \mathsf{view}^{\pi}_{\mathsf{Bob}}(a,b)\}$$

where $\stackrel{s}{\approx}$ denotes statistical indistinguishability.

The privacy requirement ensures that whatever an honest-but-curious adversary learns from interactions within the protocol can also be learned by an ideal adversary that only learns the input and output of that party.

A very useful paradigm for building private protocols is designing them in a modular way, using the following composition theorem [17]:

THEOREM 3.2. *Let f, g be two-party functionalities. Let $\pi^{f|g}$ be a private protocol for computing f using oracle calls to g and suppose that there is a private protocol π^g computing g. Let π^f be the protocol obtained from $\pi^{f|g}$ by independently using one instance of π^g for implementing each oracle call to g. Then π^f privately computes f.*

3.2 Secure Approximations

Note that the private computation of an approximation $\overline{f}$ of a target functionality f can reveal more information than the target functionality itself. Imagine, for instance, the case where the output of $\overline{f}$ is equal to the output of f in all bits except the least significant one, in which $\overline{f}$ encodes one bit of the input of the other party. To ensure that the approximation $\overline{f}$ does not leak additional information, we use the framework of Feigenbaum et al. [15, 16] for private approximations, using the notation of Kiltz et al. [23]. Only deterministic target functionalities f are considered, but the approximations $\overline{f}$ can be randomized.

DEFINITION 3.3. *The functionality $\overline{f}$ is an ε-approximation of f if for all possible inputs (a,b), $|f(a,b) - \overline{f}(a,b)| < \varepsilon$.*

DEFINITION 3.4. *The functionality $\overline{f}$ is functionally private with respect to f if there is a simulator $\mathcal{S}$ such that for all possible inputs (a,b), $\{\mathcal{S}(f(a,b))\} \stackrel{s}{\approx} \{\overline{f}(a,b)\}$.*

Note that functional privacy is a property of the functionality $\overline{f}$ itself, and not of any protocol π implementing it. It captures the fact that the approximation error is independent from the inputs when conditioned on the output of the exact functionality.

DEFINITION 3.5. *A protocol π is a private computation of an ε-approximation with respect to f if π privately computes a (possibly randomized) function $\overline{f}$ such that $\overline{f}$ is functionally private with respect to f and is an ε-approximation of f.*

3.3 Commodity Based Cryptography

In the commodity based cryptography model [3, 2], a trusted initializer (TI) distributes values (i.e., the commodities) to the parties before the start of the protocol execution. The TI has no access to the parties' secret inputs and does not communicate with the parties except for delivering the pre-distributed values during the setup. One main advantage of this model is the high computational efficiency that arises from the fact that the parties often only need to derandomize the pre-computed instances to match their own inputs. Another advantage is the computations are pre-distributed by a trusted initializer, and therefore most protocols yield perfect security. The trusted initializer functionality $\mathcal{F}_{TI}^{\mathcal{D}}$ is parametrized by an algorithm $\mathcal{D}$, which is executed upon initialization to generate the correlated randomness $(P_{\mathsf{Alice}}, P_{\mathsf{Bob}})$ that is distributed to Alice and Bob respectively.

4 Overview of Our Protocols

Assume that we have a set of training examples (real vectors)

$$(a_1(x_i), a_2(x_i), \ldots, a_m(x_i), y_i)$$

where $a_j(x_i)$ is the value of the input attribute a_j for the training example x_i $(i = 1, \ldots, n)$ and y_i is the associated output. The goal is to leverage these training examples to predict the unknown outcome for a previously unseen input as accurately as possible. To this end, we want to learn a linear function

$$y = \beta_1 a_1(x) + \beta_2 a_2(x) + \ldots + \beta_m a_m(x) + b$$

that best approximates the relation between the input variables $a_1(x), a_2(x), \ldots, a_m(x)$ and the response variable y. Throughout this paper we assume that all variables are real numbers and that we aim to find real values for the parameters $\beta_1, \beta_2, \ldots, \beta_m$ and b that minimize the empirical risk function

$$\frac{1}{n}\sum_{i=1}^{n}((\beta_1 a_1(x_i) + \beta_2 a_2(x_i) + \ldots + \beta_m a_m(x_i) + b) - y_i)^2 \quad (1)$$

which is the mean squared error over the training instances. For notational convenience, we switch to the homogenous version of the linear function and we use vector notation, i.e. let

- $\mathbf{x_i} = (a_0(x_i), a_1(x_i), a_2(x_i), \ldots, a_m(x_i))$, with $a_0(x_i) = 1$ for all $i \in \{1, \ldots, n\}$
- $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_m)$, with $\beta_0 = b$

Using $\langle \boldsymbol{\beta}, \mathbf{x_i} \rangle$ to denote the dot product of $\boldsymbol{\beta}$ and $\mathbf{x_i}$, minimizing (1) amounts to calculating the gradient and comparing it to zero, i.e. solving

$$\frac{2}{n} \sum_{i=1}^{n} (\langle \boldsymbol{\beta}, \mathbf{x_i} \rangle - y_i) \mathbf{x_i} = 0 \tag{2}$$

The solution to (2) is[1]

$$\boldsymbol{\beta} = (X^T X)^{-1} X^T \mathbf{y} \tag{3}$$

with

$$X = \begin{pmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \cdots \\ \mathbf{x_n} \end{pmatrix} \text{ and } \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{pmatrix}$$

The scenarios that we are interested in are those in which the training data is not owned by a single party but is instead distributed across multiple parties who are not willing to disclose it. Our experiments in Section 10 correspond to scenarios in which X is partitioned column-wise across two parties, i.e. Alice and Bob have information about different features of the same instances. However, as will become clear below, our protocols work in all scenarios in which Alice has a share X_{Alice} and Bob has a share X_{Bob} such that $X_{\mathsf{Alice}} + X_{\mathsf{Bob}} = X$, regardless of whether the dataset X is sliced column-wise, row-wise, or a mixture of the two. In our experiments we also assume that Bob has the vector Y. However, our protocol can also handle the case when Y is distributed over two or more players.

Here we give an overview of our solution. The basic idea is to reduce the problem of securely computing linear regression to the problem of securely computing products of matrices. The protocol for computing products of matrices works only for elements of the matrices belonging to a finite field. Thus, Alice and Bob should be able to map their real valued fixed precision inputs to elements of a finite field (as described in Section 6). Our protocol works in a shared input model in which each party holds some elements of the design matrix. Each party creates its share of the design matrix by mapping their respective real valued inputs to elements of a finite field and putting them on the respective position of the matrix and then filling the remaining positions of the matrix's share with zeros. I.e., the shares X_{Alice} and X_{Bob} are such that $X_{\mathsf{Alice}} + X_{\mathsf{Bob}} = X$ where X is the design matrix mapped into the finite field.

1. Offline phase: in the information-theoretically secure protocol, Alice and Bob receive correlated data from the Trusted Initializer. In the case of the computationally secure protocol, they run the protocol described in Section 9.
2. Online Phase:
   (a) The players map their fixed precision real valued inputs to elements of a finite field as described in Section 6 and create the shares of X as described above.
   (b) The players compute over their shares using the protocols for matrix multiplication (described in Section 5) and for computing the inverse of a Covariance Matrix (described in Section 7) in order to obtain shares of the estimated regression coefficient vector.
   (c) The players exchange their shares of the estimated regression coefficient vector and reconstruct it.

[1] Assuming that $X^T X$ is invertible

After presenting the building blocks in Sections 5, 6 and 7, we reiterate the information-theoretically secure and the computationally secure protocol for linear regression at a more concrete level of detail in Sections 8 and 9 respectively.

In presenting our protocols, we first introduce an ideal functionality that captures the behaviour of a secure instance of the protocol in question. Ideal functionalities are always represented by calligraphic letters (e.g. $\mathcal{F}_{\mathsf{DMM}}$ in the case of distributed the matrix multiplication functionality). We then present the protocol and prove that the protocol is as secure as the ideal functionality. Protocols are represented by capital greek letters (e.g. Π_{DMM} in the case of the distributed matrix multiplication protocol).

5 Secure Distributed Matrix Multiplication Protocol

Let's first take a look at a straightforward extension of Beaver's protocol for secure multiplication in the commodity based model [4] for matrices. Alice and Bob hold shares of matrices $X \in \mathbb{Z}_q^{n_1 \times n_2}$ and $Y \in \mathbb{Z}_q^{n_2 \times n_3}$ to be multiplied and the goal is to obtain shares of the multiplication result $X \cdot Y \in \mathbb{Z}_q^{n_1 \times n_3}$ in such a way that the result remains hidden from both of them individually. Let $X_{\mathsf{Alice}} \in \mathbb{Z}_q^{n_1 \times n_2}$ and $Y_{\mathsf{Alice}} \in \mathbb{Z}_q^{n_2 \times n_3}$ be Alice's shares of the inputs and $X_{\mathsf{Bob}} \in \mathbb{Z}_q^{n_1 \times n_2}$ and $Y_{\mathsf{Bob}} \in \mathbb{Z}_q^{n_2 \times n_3}$ be Bob's shares. Note that $XY = (X_{\mathsf{Alice}} + X_{\mathsf{Bob}})(Y_{\mathsf{Alice}} + Y_{\mathsf{Bob}}) = X_{\mathsf{Alice}}Y_{\mathsf{Alice}} + X_{\mathsf{Alice}}Y_{\mathsf{Bob}} + X_{\mathsf{Bob}}Y_{\mathsf{Alice}} + X_{\mathsf{Bob}}Y_{\mathsf{Bob}}$. The terms $X_{\mathsf{Alice}}Y_{\mathsf{Alice}}$ and $X_{\mathsf{Bob}}Y_{\mathsf{Bob}}$ can be computed locally, but the computation of the terms $X_{\mathsf{Alice}}Y_{\mathsf{Bob}}$ and $X_{\mathsf{Bob}}Y_{\mathsf{Alice}}$ is more complex. Beaver's protocol solves the problem of computing the last two terms by having the trusted initializer distribute random values $A_{\mathsf{Alice}}, A_{\mathsf{Bob}}$, $B_{\mathsf{Alice}}, B_{\mathsf{Bob}}$ to the parties and also random shares of the value $A_{\mathsf{Alice}}B_{\mathsf{Bob}} + A_{\mathsf{Bob}}B_{\mathsf{Alice}}$. Then the parties only need to derandomize this pre-distributed instance to the actual input values. The protocol Π_{DMM} is parametrized by the size q of the field and by the dimensions n_1, n_2, n_3 of the matrices to be multiplied and works as follows:

1. At the setup, the trusted initializer chooses uniformly random $A_{\mathsf{Alice}}, A_{\mathsf{Bob}} \in \mathbb{Z}_q^{n_1 \times n_2}$, $B_{\mathsf{Alice}}, B_{\mathsf{Bob}} \in \mathbb{Z}_q^{n_2 \times n_3}$ and $T \in \mathbb{Z}_q^{n_1 \times n_3}$, and distributes the values $A_{\mathsf{Alice}}, B_{\mathsf{Alice}}$, T to Alice and the values $A_{\mathsf{Bob}}, B_{\mathsf{Bob}}, C = (A_{\mathsf{Alice}}B_{\mathsf{Bob}} + A_{\mathsf{Bob}}B_{\mathsf{Alice}} - T)$ to Bob.
2. Bob sends $(X_{\mathsf{Bob}} - A_{\mathsf{Bob}})$ and $(Y_{\mathsf{Bob}} - B_{\mathsf{Bob}})$ to Alice.
3. Alice chooses a random $T' \in \mathbb{Z}_q^{n_1 \times n_3}$, computes $W = A_{\mathsf{Alice}}(Y_{\mathsf{Bob}} - B_{\mathsf{Bob}}) + (X_{\mathsf{Bob}} - A_{\mathsf{Bob}})B_{\mathsf{Alice}} + X_{\mathsf{Alice}}Y_{\mathsf{Alice}} - T'$ and sends W, $(X_{\mathsf{Alice}} - A_{\mathsf{Alice}})$ and $(Y_{\mathsf{Alice}} - B_{\mathsf{Alice}})$ to Bob. Alice outputs $T + T'$.
4. Bob outputs $U = (X_{\mathsf{Alice}} - A_{\mathsf{Alice}})Y_{\mathsf{Bob}} + X_{\mathsf{Bob}}(Y_{\mathsf{Alice}} - B_{\mathsf{Alice}}) + X_{\mathsf{Bob}}Y_{\mathsf{Bob}} + W + C$.

This protocol securely implements the ideal distributed matrix multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ that works as follows: $\mathcal{F}_{\mathsf{DMM}}$ is parametrized by the size q of the field and the dimensions n_1, n_2, n_3 of the matrices to be multiplied. Given Alice's input shares $X_{\mathsf{Alice}} \in \mathbb{Z}_q^{n_1 \times n_2}$ and $Y_{\mathsf{Alice}} \in \mathbb{Z}_q^{n_2 \times n_3}$, and Bob's input shares $X_{\mathsf{Bob}} \in \mathbb{Z}_q^{n_1 \times n_2}$ and $Y_{\mathsf{Bob}} \in \mathbb{Z}_q^{n_2 \times n_3}$, it computes $V = (X_{\mathsf{Alice}} + X_{\mathsf{Bob}})(Y_{\mathsf{Alice}} + Y_{\mathsf{Bob}})$, chooses a random matrix $R \in \mathbb{Z}_q^{n_1 \times n_3}$ and sends R to Alice and $V - R$ to Bob.

THEOREM 5.1. *The distributed matrix multiplication protocol Π_{DMM} is correct and securely implements the distributed matrix multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ against honest but curious adversaries in the commodity based model.*

The correctness of the protocol can be trivially verified by inspecting the value of $T + T' + U$. The security of this protocol lies in the fact that all values exchanged between the parties are blinded by a value which is completely random in the underlying field from the point of view of the message receiver. This protocol can in fact be proved secure even against malicious parties and in the stronger Universal Composability (UC) framework [8]. The idea is that the simulator simulates an instance of the adversary and an instance of the protocol execution with the adversary, allowing the adversary to communicate with the environment. The leverage used by the simulator is the fact that, in the ideal execution, he is the one simulating the trusted initializer. By simulating the TI, he is able, at the same time, to generate a protocol transcript for the adversary that is indistinguishable from the real protocol execution and also to extract the input of the corrupted parties in order to forward them to the ideal functionality.

6 Dealing with Real Numbers

The security proof of the (matrix) multiplication protocol presented in the last section essentially relies on the fact that the blinding factors are uniformly random in $\mathbb{Z}_q$. If one tries to design similar protocols working directly with integers or real numbers there would be a problem, since it is not possible to sample uniformly in $\mathbb{Z}$ or $\mathbb{R}$. Similarly, in protocols that use homomorphic encryption as building blocks, the encryption is normally done for messages which are members of a finite group. But in secure protocols for functionalities such as linear regression, one needs to deal with inputs which are real numbers. Thus it is necessary to develop a way to approximate the computations on real numbers by using building blocks which work on fields $\mathbb{Z}_q$.

We adapt the method of Catrina and Saxena [9] with a fixed-point representation. Let k, e and f be integers such that $k > 0$, $f \geq 0$ and $e = k - f \geq 0$. Let $\mathbb{Z}_{\langle k \rangle}$ denote the set $\{x \in \mathbb{Z} : -2^{k-1} + 1 \leq x \leq 2^{k-1} - 1\}$. The fixed-point data type with k bits, resolution 2^{-f}, and range 2^e is the set $\mathbb{Q}_{\langle k,f \rangle} = \{\tilde{x} \in \mathbb{Q} : \tilde{x} = \hat{x}2^{-f}, \hat{x} \in \mathbb{Z}_{\langle k \rangle}\}$. The signed integers in $\mathbb{Z}_{\langle k \rangle}$ are then encoded in the field $\mathbb{Z}_q$ (with $q > 2^k$) using the function

$$g\colon \mathbb{Z}_{\langle k \rangle} \to \mathbb{Z}_q, g(\hat{x}) = \hat{x} \mod q.$$

In secure computation protocols using secret sharing techniques, the values in $\mathbb{Z}_q$ are actually shared between the two parties. Using this encoding, we have that $\hat{x} + \hat{y} = g^{-1}(g(\hat{x}) + g(\hat{y}))$, where the second addition is in $\mathbb{Z}_q$, i.e., we can compute the addition for signed integers in $\mathbb{Z}_{\langle k \rangle}$ by using the arithmetic in $\mathbb{Z}_q$. The same holds for subtraction and multiplication.

For the fixed-point data type we can do additions using the fact that $\tilde{w} = \tilde{x} + \tilde{y} = (\hat{x} + \hat{y})2^{-f}$, so we can trivially obtain the representation of $\tilde{w}$ with resolution 2^{-f} by computing $\hat{w} = \hat{x} + \hat{y}$, i.e., we can do the addition of the fixed-point type by using the addition in $\mathbb{Z}_{\langle k \rangle}$, which itself can be done by performing the addition in $\mathbb{Z}_q$. The same holds for subtraction. But for multiplication we have that $\tilde{w} = \tilde{x}\tilde{y} = \hat{x}\hat{y}2^{-2f}$, and therefore if we perform the multiplication in $\mathbb{Z}_q$, we will obtain (if no overflow occurred) the representation of $\tilde{w}$ with resolution 2^{-2f}. Such representation uses $\mathbb{Z}_{\langle k+f \rangle}$ instead of the original $\mathbb{Z}_{\langle k \rangle}$. In order to have the size of the signed integers representation be independent from the amount of multiplication operations performed with the fixed-point data, we need to scale the resolution of $\tilde{w}$ down to 2^{-f}. For that purpose we use a slightly modified version of the truncation protocol of Catrina and Saxena [9].

The central idea of this truncation protocol is to reveal the number w to be truncated to one of the parties, but blinded by a factor r which is from a domain exponentially bigger than the domain of the value w and thus statistically hides w. The value r is generated so that the parties have shares of both r and r' (which represents the f least significant bits of r). Then Bob can reveal $w + r$ to Alice and they can compute shares of the truncated value by using local computations. In more detail, let λ be a security parameter and let the field $\mathbb{Z}_q$ in which the signed integers are encoded be such that $q > 2^{k+f+\lambda+1}$. Note that the multiplicative inverse of 2^f in $\mathbb{Z}_q$ is $((q+1)/2)^f$ and let F^{-1} denote it. The parties have, as inputs, shares $w_{\mathsf{Alice}}, w_{\mathsf{Bob}} \in \mathbb{Z}_q$ such that $w = (w_{\mathsf{Alice}} + w_{\mathsf{Bob}}) \in \{0, 1, \ldots, 2^{k+f-1} - 1\} \cup \{q - 2^{k+f-1} + 1, \ldots, q - 1\}$. The protocol Π_{Trunc} for truncating the output works as follows:

1. At the setup, the trusted initializer chooses uniformly random $r' \in \{0,1\}^f$ and $r'' \in \{0,1\}^{\lambda+k}$ and computes $r = r''2^f + r'$. He also chooses uniformly random $r'_{\mathsf{Bob}}, r_{\mathsf{Bob}} \in \mathbb{Z}_q$ and then sets $r'_{\mathsf{Alice}} = r' - r'_{\mathsf{Bob}}$ and $r_{\mathsf{Alice}} = r - r_{\mathsf{Bob}}$. He sends $r'_{\mathsf{Alice}}, r_{\mathsf{Alice}}$ to Alice and $r'_{\mathsf{Bob}}, r_{\mathsf{Bob}}$ to Bob.

2. Bob sends $z_{\mathsf{Bob}} = (w_{\mathsf{Bob}} + r_{\mathsf{Bob}})$ to Alice and outputs $(w_{\mathsf{Bob}} + r'_{\mathsf{Bob}})F^{-1}$.

3. Alice computes $c = z_{\mathsf{Bob}} + w_{\mathsf{Alice}} + r_{\mathsf{Alice}} + 2^{k+f-1}$ and $c' = c \mod 2^f$. Then she outputs $(w_{\mathsf{Alice}} + r'_{\mathsf{Alice}} - c')F^{-1}$.

This protocol securely implements the functionality $\mathcal{F}_{\mathsf{Trunc}}$ that captures the approximate truncation without leakage. $\mathcal{F}_{\mathsf{Trunc}}$ is parametrized by the size q of the field and reduces the resolution by 2^{-f}. Given Alice and Bob's shares of the input, $w_{\mathsf{Alice}}, w_{\mathsf{Bob}} \in \mathbb{Z}_q$, and a blinding factor $r'_{\mathsf{Bob}} \in \mathbb{Z}_q$ from Bob, it computes $\hat{w} = g^{-1}(w_{\mathsf{Alice}} + w_{\mathsf{Bob}} \mod q)$ and samples u such that $\Pr[u = 1] = (\hat{w} \mod 2^f)/2^f$. Then it computes $v = (w_{\mathsf{Alice}} - (w_{\mathsf{Alice}} + w_{\mathsf{Bob}} \mod 2^f) - r'_{\mathsf{Bob}})F^{-1} + u$ and sends it to Alice (Bob's output is $(w_{\mathsf{Bob}} + r'_{\mathsf{Bob}})F^{-1}$ and can be locally computed).

THEOREM 6.1. *The truncation protocol Π_{Trunc} privately computes the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$.*

PROOF. **Correctness**: Let $\hat{w} = g^{-1}(w_{\mathsf{Alice}} + w_{\mathsf{Bob}} \mod q)$. We have that the value $\hat{w} \in \{-2^{k+f-1} + 1, -2^{k+f-1} + 2, \ldots, 2^{k+f-1} - 1\}$. Let $b = \hat{w} + 2^{k+f-1}$ and let $b' = b$

mod 2^f. We have that $b \in \{1, \ldots, 2^{k+f}-1\}$ and since $k > 0$ also that

$$b' = b \mod 2^f = \hat{w} + 2^{k+f-1} \mod 2^f = \hat{w} \mod 2^f.$$

Since $r < 2^{k+f+\lambda}$ and $q > 2^{k+f+\lambda+1}$ we have that $c = b + r$ and thus

$$c' = (b' + r') \mod 2^f = b' + r' - u2^f$$

where $u \in \{0,1\}$ and $\Pr[u = 1] = \Pr[r' \geq 2^f - b']$ $= (\hat{w} \mod 2^f)/2^f$ with the probability over the choices of random r'. Hence

$$c' - r'_{\mathsf{Alice}} - r'_{\mathsf{Bob}} = g(\hat{w} \mod 2^f - u2^f),$$

$$w_{\mathsf{Alice}} + w_{\mathsf{Bob}} + r'_{\mathsf{Alice}} + r'_{\mathsf{Bob}} - c' = g(\hat{w} - (\hat{w} \mod 2^f) + u2^f)$$

$$= g\left(\left\lfloor \frac{\hat{w}}{2^f} \right\rfloor 2^f + u2^f\right),$$

$$(w_{\mathsf{Alice}} + w_{\mathsf{Bob}} + r'_{\mathsf{Alice}} + r'_{\mathsf{Bob}} - c')F^{-1} = g\left(\left\lfloor \frac{\hat{w}}{2^f} \right\rfloor + u\right),$$

and so the shares output by the parties $(w_{\mathsf{Alice}} + r'_{\mathsf{Alice}} - c')F^{-1}$ and $(w_{\mathsf{Bob}} + r'_{\mathsf{Bob}})F^{-1}$ are correct.

Security: The only message exchanged is $z_{\mathsf{Bob}} = (w_{\mathsf{Bob}} + r_{\mathsf{Bob}})$ that reveals $c = b + r$ to Alice, but since r is a uniformly random $(k+f+\lambda)$-bits integer and b is a $(k+f)$-bits integer, we have that the statistical distance between c and r is at most $2^{-\lambda}$, i.e., c is statistically indistinguishable from a uniformly random value. □

THEOREM 6.2. *$\mathcal{F}_{\mathsf{Trunc}}$ is an 1-approximation[2] and is functionally private with respect to an exact truncation functionality that computes the truncation using the floor function.*

PROOF. The only difference between the two functionalities is that in the approximate truncation an error factor u is present in the shared output. But note that $u \in \{0,1\}$ and $\Pr[u = 1] = (\hat{w} \mod 2^f)/2^f$, but u is independent from the specific shares w_{Alice}, w_{Bob} used to encode $g(\hat{w})$. Thus the protocol rounds $\hat{w}/2^f$ to the nearest integer with probability $1 - \alpha$, where α is the distance between $\hat{w}/2^f$ and the nearest integer. □

We should mention that in the case of matrix multiplication the truncations only have to be performed in the elements of the resulting matrix instead of once per element multiplication, which would be less efficient and also increase the error due to truncation.

7 Computing the Inverse of a Covariance Matrix

In order to be able to compute the linear regression model from a design matrix and the response vector we need to compute the inverse of the covariance matrix. Let A be the covariance matrix. In order to compute A^{-1} we use a generalization for matrices of the Newton-Raphson division method.

The algorithms for division of fixed-point numbers are divided in two main classes: digit recurrence (subtractive division) and functional iteration (multiplicative division). The Newton-Raphson division method is from the functional iteration class, which is more amenable to secure implementation and converges faster. Additionally this method is self correcting, i.e., truncation errors in one iteration decrease quadratically in the next iterations. The inverse of a number a is computed by defining the function $h(x) = x^{-1} - a$ and then applying the Newton-Raphson method for finding successively better approximations to the roots of $h(x)$. The iterations follow the form:

$$x_{s+1} = x_s(2 - ax_s).$$

This algorithm can be generalized for computing the inverse of the matrix A. A numerical stable iteration for computing A^{-1} is [19, 18]:

$$\begin{aligned} c &= \mathsf{trace}(A) \\ X_0 &= c^{-1}I \\ X_{s+1} &= X_s(2 - AX_s) \end{aligned}$$

where I is the identity matrix with the same dimensions as A. Note that A is a covariance matrix and thus it is positive semi-definite and the trace of A dominates the largest eigenvalue of A. It is convenient to use $c = \mathsf{trace}(A)$ because the trace of A can be computed locally by parties that have shares of A. To compute c^{-1} the Newton-Raphson is also used with x_0 set to an arbitrarily small value, as the convergence happens if the magnitude of the initial value is smaller than that of c^{-1}.

Note that, in our case, we use this method to securely compute the inverse of the covariance matrix, i.e, each party has a share of the covariance matrix as input and should receive, as output, random shares of its inverse, but no additional information should be learned by the parties. Hence we cannot perform a test after each iteration in order to check if the values already converged with resolution 2^{-f} (and thus stop the iterations at the optimal point) because this would leak information about the input based on how many iterations were performed. We have to use an upper bound ℓ on the number of iterations such that all possible covariance matrices converge with resolution 2^{-f} in ℓ iterations. A very conservative upper bound is $\ell = 2k$ [19]; further studies will be done to try to determine a tighter upper bound.

The protocol to securely compute the inverse of a shared covariance matrix is parametrized by the size q of the field. Let $A \in \mathbb{Z}_q^{n \times n}$ be the encoding in $\mathbb{Z}_q$ of a covariance matrix where the elements are fixed-point numbers. Alice has input $A_{\mathsf{Alice}} \in \mathbb{Z}_q^{n \times n}$ and Bob has input $A_{\mathsf{Bob}} \in \mathbb{Z}_q^{n \times n}$ such that $A_{\mathsf{Alice}} + A_{\mathsf{Bob}} = A$. Then the protocol proceeds as follows:

1. Alice and Bob locally compute shares of $c = \mathsf{trace}(A)$, i.e., $c_{\mathsf{Alice}} = \sum_{i=1}^{n} A_{\mathsf{Alice}}[i,i]$ and $c_{\mathsf{Bob}} = \sum_{i=1}^{n} A_{\mathsf{Bob}}[i,i]$
2. Alice and Bob use the Newton-Raphson division method to compute shares c_{Alice}^{-1} and c_{Bob}^{-1} of c^{-1} with resolution 2^{-f}. The subtractions can be performed locally and the multiplications using the distributed (matrix) multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ followed by the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$.
3. Alice and Bob use the generalized Newton-Raphson method to obtain shares A_{Alice}^{-1} and A_{Bob}^{-1} of A^{-1} with resolution 2^{-f} for the elements. The subtractions can be performed locally and the multiplications using the distributed matrix functionality $\mathcal{F}_{\mathsf{DMM}}$ followed by the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$.

[2] in the representation, 2^{-f} in the fixed-point data type

We emphasize that the truncation used has some intrinsic error, but the Newton-Raphson method's self-correcting properties compensate for that.

8 Computing the Linear Regression Coefficients

We consider the setting in which there is a design matrix $\tilde{X}$ and a response vector $\tilde{y}$. We are interested in analyzing the statistical regression model

$$\tilde{y} = \tilde{X}\tilde{\beta} + \epsilon,$$

and therefore want to compute the estimated regression coefficient vector

$$\overline{\beta} = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T\tilde{y}$$

The design matrix is a $n \times m$ matrix where the elements are of the fixed-point data type with precision 2^{-f} and range 2^{k-f} (i.e., $\tilde{X} \in \mathbb{Q}^{n\times m}_{\langle k,f\rangle}$) and the response vector $\tilde{y} \in \mathbb{Q}^{n}_{\langle k,f\rangle}$. Let $\hat{X} \in \mathbb{Z}^{n\times m}_{\langle k\rangle}$ be the element-wise representation of $\tilde{X}$ as signed integers and let $X \in \mathbb{Z}^{n\times m}_q$ be the element-wise encoding of $\hat{X}$ as elements of the field $\mathbb{Z}_q$. Define $\hat{y}$ and y in the same way from $\tilde{y}$.

It is assumed that the parties Alice and Bob hold shares of X and y. Alice and Bob can then use the protocols for matrix multiplication, truncation and covariance matrix inversion that were described in the previous sections in order to compute shares of

$$\beta = (X^TX)^{-1}X^Ty$$

Then they only need to reveal their final shares and convert the result back to the fixed-point data type in order to get $\overline{\beta}$.

In more details:

1. Online Phase:
   (a) The players map their fixed precision real valued inputs to elements of a finite field as described in Section 6 and create the shares of X as described above.
   (b) The players compute X^TX by using the matrix multiplication protocol Π_{DMM} (described in Section 5). Once the multiplication is finished they ran the truncation protocol Π_{Trunc} for each element of the matrix X^TX.
   (c) Alice and Bob compute the inverse of (X^TX) by running the protocol for computing the inverse of a covariance matrix (described in Section 7). Within the covariance matrix inversion protocol there are several calls to the matrix multiplication and truncation protocols.
   (d) Alice and Bob run the matrix multiplication and the truncation protocols twice to obtain $(X^TX)^{-1}X^T$ and finally $(X^TX)^{-1}X^Ty$.
   (e) The players exchange their shares of the estimated regression coefficient vector and reconstruct it.
   (f) The coefficients β obtained by the players are mapped back from finite field elements to real values with finite precision.

The security of the composed protocol follows from the composition theorem 3.2 using the facts that Π_{DMM} securely implements the distributed matrix multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ and Π_{Trunc} privately computes the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$. It is assumed that a big enough k is used so that no overflow occurs and hence the correctness of the protocol follows. The final protocol implements the linear regression functionality $\mathcal{F}_{\mathsf{Reg}}$ that upon getting the shares X_{Alice} and X_{Bob} of the design matrix X and y_{Alice} and y_{Bob} of the response vector y, compute $\beta = (X^TX)^{-1}X^Ty$ and output β to Alice and Bob.

9 Substituting the Trusted Initializer

If a trusted initializer is not desired, it is possible to obtain a solution where the parties, during the setup phase, compute the correlated data themselves. The idea is to use the homomorphic properties of the Paillier's encryption scheme [27]. For two large prime numbers p and q, the secret key of Paillier's cryptosystem is $\mathsf{sk} = (p, q)$. The corresponding public key is $\mathsf{pk} = N = pq$ and the encryption of a message $x \in \mathbb{Z}_N$ is done by picking a random $r \in \mathbb{Z}^*_{N^2}$ and computing $\mathsf{Enc}(\mathsf{pk}, x) = (N+1)^x r^N \mod N^2$. The following homomorphic properties of Paillier's encryption scheme are used:

$$\mathsf{Enc}(\mathsf{pk}, x) \cdot \mathsf{Enc}(\mathsf{pk}, y) = \mathsf{Enc}(\mathsf{pk}, x + y \mod N)$$

$$\mathsf{Enc}(\mathsf{pk}, x)^y = \mathsf{Enc}(\mathsf{pk}, xy \mod N).$$

Given two vectors $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ where the second is given in clear and the first is encrypted element-wise (i.e., $\mathsf{Enc}(\mathsf{pk}, x_i)$ are revealed), one can compute a ciphertext corresponding to the inner product:

$$\mathsf{Enc}(\mathsf{pk}, \langle x, y\rangle \mod N) = \prod_{i=1}^{n} \mathsf{Enc}(\mathsf{pk}, x_i)^{y_i}.$$

The idea for computing the necessary correlated data for the distributed matrix multiplication protocol is to use the above fact in order to compute the non-local multiplication terms. Bob has a pair of public/secret keys for Paillier's encryption scheme and sends to Alice the element-wise encryption *under his own public key* of the elements of the column/row that needs to get multiplied. Alice, having the plaintext corresponding to her own values on the appropriate column/row, can compute an encryption of the inner product under Bob's public key. She then adds a random blinding factor and sends the ciphertext to Bob, who can decrypt it, thus yielding distributed shares of the inner product between Alice and Bob.

The protocol is parametrized by the dimensions n_1, n_2, n_3 of the matrices to be multiplied. Bob holds a Paillier's secret key sk, whose corresponding public-key is pk. For a matrix X, $x[i, j]$ denote the element in the i-th row and j-th column. The protocol works as follows:

1. Alice chooses uniformly random $A_{\mathsf{Alice}} \in \mathbb{Z}_N^{n_1\times n_2}$, $B_{\mathsf{Alice}} \in \mathbb{Z}_N^{n_2\times n_3}$ and $T \in \mathbb{Z}_N^{n_1\times n_3}$.
2. Bob chooses uniformly random $A_{\mathsf{Bob}} \in \mathbb{Z}_N^{n_1\times n_2}$ and $B_{\mathsf{Bob}} \in \mathbb{Z}_N^{n_2\times n_3}$, element-wise encrypts them under his own public key and send the ciphertexts to Alice.

3. For $i = 1, \ldots, n_1$, $j = 1, \ldots, n_3$, Alice computes the ciphertext

$$\tilde{c}[i,j] = \mathsf{Enc}\left(\mathsf{pk}, t[i,j]\right) \cdot \prod_{k=1}^{n_2} \Big(\mathsf{Enc}\left(\mathsf{pk}, b_{\mathsf{Bob}}[k,j]\right)^{a_{\mathsf{Alice}}[i,k]} \cdot \cdot \mathsf{Enc}\left(\mathsf{pk}, a_{\mathsf{Bob}}[i,k]\right)^{b_{\mathsf{Alice}}[k,j]} \Big)$$

and sends them to Bob. Alice outputs A_{Alice}, B_{Alice} and T.

4. Bob decrypts the ciphertexts in order to get the matrix $C = (A_{\mathsf{Alice}} B_{\mathsf{Bob}} + A_{\mathsf{Bob}} B_{\mathsf{Alice}} + T)$. Bob outputs A_{Bob}, B_{Bob} and C.

The security of this protocol follows trivially from the IND-CPA security of Paillier's encryption scheme [27].

Note that the values r and r' that are distributed by the trusted initializer for performing the truncation protocol can be trivially computed by the parties themselves using distributed multiplications.

10 Experiments

We assessed our secure linear regression algorithm by implementing it and analyzing the results using ten real datasets. We chose a variety of different datasets based on the number of features and the number of instances (see Section 10.1). We used C++ as our programming language which we augmented with the BOOST libraries for functionality such as lexical_cast for type casting and asio for work with sockets. We also made use of the GMP and NTL libraries within C++ to implement our protocols. We built our system on top of a Microsoft Azure G4 series machine with Intel Xeon processor E5 v3 family, 224GB RAM size, 3072GB of disk size and 16 cores. Finally, we chose Ubuntu 12.04 as our operating system. We have merged the matrix multiplication and truncation protocols within one protocol for implementation purposes.

The online phase (Section 10.2) is very fast and capable of handling millions of records within less than an hour, which is a huge improvement to the previous results. We only use addition and multiplication of matrices on our online phase which makes it simple and easy to manage.

In the case when a trusted initializer is not desired one can use our computationally secure protocol, at the cost of having a costier offline phase (Section 10.3). However, because Alice and Bob only work over random inputs during the offline phase, the encryption, decryption and mathematical operations are all embarassingly parallelizable.

10.1 Datasets

All our datasets are contained within the UCI repository[3], with the exception of the State Inpatient Database (WA) which is provided by HCUP[4]. The UCI repository includes 48 regression task datasets from which we chose 9. Our datasets range in size from 395 instances to over 4 million and from 7 attributes to 367, and are summarized in Table 1.

[3]UC Irvine Machine Learning Repository
`https://archive.ics.uci.edu/ml/datasets.html`
[4]`http://www.ahrq.gov/research/data/hcup/`

Gas Sensor Array Under Dynamic Gas Mixtures This dataset represents data from 16 chemical sensors exposed to ethylene and CO mixtures at various concentration levels in the air. We added together the concentration of ethylene and the concentration of CO to create one continuous response variable of gas concentration and removed the time feature from the dataset. We then designated the first 8 sensor readings to Alice and the second 8 to Bob. This left us with a total of 4,208,261 sets of 16 sensor readings to different total concentrations of ethylene and CO.

Communities and Crime We used 122 attributes describing 1,993 communities and their law enforcement departments in 1990 to create this dataset. The goal with this dataset is to predict the number of violent crimes per capita for each community. All missing values present in the dataset (of which there were 36,850 distributed throughout 1,675 different communities and 22 different attributes) were replaced with 0s. These missing values were largely relevant to the communities' police departments. We also removed 5 variables that were present in the original data but described by the UCI documentation as non-predictive, namely state, county, community, community name, and fold. The final 122 attributes were then divided in half between Alice and Bob.

Auto MPG This dataset contains attributes describing 398 automobiles in attempt to predict MPG (miles per gallon) for each. We removed the car name attribute which was present in the original data and were left with 7 predictive features. We then replaced the 6 missing horsepower values with 0s. In the end we designated the cylinders, displacement, and horsepower features to Alice and the weight, acceleration, model year, and origin features to Bob.

BlogFeedback In an attempt to predict the number of comments a blog post will receive in the upcoming 24 hours, this dataset originally had 280 attributes. Since our complete dataset must be linearly independent, to enable the inversion of $X^T X$ required in our protocol, we removed 57 of these original attributes leaving us with 223 predictors describing 52,397 blog posts. An example of such a feature would be the binary indicator of whether or not a blog post was published on a Sunday. There are binary indicators of whether publication occurred on any of the other days of the week and therefore this feature, publication on a Sunday, is linearly dependent on the other six. Finally, the dataset was divided column wise, designating 111 attributes to Alice and the other 112 to Bob.

Wine Quality This dataset takes 11 attributes related to the white variant of Portuguese "Vinho Verde" wine which are used to predict the quality score of 4,897 wines. We designated the fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, and free sulfur dioxide features to Alice and the total sulfur dioxide, density, pH, sulphates, and alcohol features to Bob.

Bike Sharing In this dataset we took attributes describing a certain hour and day and attempted to predict the number of users for a bike sharing system. We removed

the record index which was present in the original data as well as the count of casual users and the count of registered users and targeted the total rental bikes used ($casual + registered$) for our prediction. We were left with 13 predictors of 17,379 hour/day combinations which were used to model bike use. Alice received information on the dates, seasons, years, months, hours, and holidays while Bob was given information on weekdays, working days, weather situations, temperatures, feel temperatures, humidities, and windspeeds.

Student Performance We used 30 attributes describing 395 students across two schools to create this dataset. The goal with this dataset is to predict the final grade of each student in their math class. We removed two columns from the original dataset – one detailing students' performances in the first period and one detailing their performances in the second period. We identified the student's final grade as our sole response variable. The final 30 attributes were then divided evenly between Alice and Bob.

YearPredictionMSD In this dataset we have attributes describing audio features of 515,344 songs and we aim to predict the release year of each song. We kept all 90 features that were present in the original data provided by the UCI repository. In allocating the data we gave Alice the first 45 features and the second 45 to Bob.

State Inpatient Database (WA) From the HCUP State Inpatient Database (WA) we extracted attributes describing 25,180 beneficiaries who had at least one hospital admission within the state of Washington during the first nine months of the year between the years 2009 and 2012. The goal with this data is to predict the cost each beneficiary will incur in the final three months of the same year. We extracted demographic, medical, and previous cost information from the original data and replaced any missing values with a 0 value. We then designated the age, gender, race, number of chronic conditions, length of stay, and number of admits attributes to Alice. Bob was given a Boolean matrix of comorbidities as well as previous cost information.

Relative Location of CT Slices on Axial Axis In an attempt to predict the relative location of a CT slice on the axial axis of the human body, the original dataset had 384 attributes describing CT images. Since our complete dataset must be linearly independent, we removed 17 of these original attributes leaving us with 367 predictors describing 53,500 CT images. We then divided this dataset column wise, designating 183 attributes to Alice and the other 184 to Bob.

10.2 Online Phase

We present in Table 1 the running times for the online phase of our protocol building a predictive linear regression model. Our online phase is very fast, computing a linear regression model for a matrix of over 4 million rows and 16 columns in under one hour. The regression coefficients computed with our secure protocol agree to the 5th decimal digit with regression coefficients computed without any security.

We briefly work out the theoretical computational complexity of computing $\boldsymbol{\beta} = (X^T X)^{-1} X^T \mathbf{y}$ with our online protocol. If our dataset (which is denoted by X in this formula), has n features and m records, then the total runtime for computing the $\boldsymbol{\beta}$ values is $O(mn^2)$ which means that the number of records in the dataset has only a linear effect on the run time of our implementation.

We used NTL for matrix multiplication with modular arithmetic. We also used GMP (the GNU Multi-Precision library) in conjunction with NTL to increase our performance. In the NTL library, the basic algorithm is used (with time complexity $O(n^3)$ when all matrix dimensions are n).

Note that $(X^T X)$ is a square matrix with both dimensions equal to n, and for datasets in which the number of features is small relative to the number of records, computing $(X^T X)^{-1}$ is very fast and negligible in respect to, for example, computing $X^T X$. Our online phase is faster and independent from the semi-honest trusted party unlike similar implementations, such as Nikolaenko et al.'s implementation [26].

10.3 Computationally Secure Offline Phase

In the pre-processing of the computationally secure offline phase of the matrix multiplication protocol Π_{DMM}, we use Paillier for encryption and decryption, but any additive homomorphic encryption scheme can be used. The down side of these schemes is that their encryption and decryption times are computationally intensive and, if the given dataset is large, the pre-processing phase can take a long time. This issue can be tackled by noticing that Alice and Bob, during this phase, only work over random inputs and thus one can use heavy parallelization to speed-up the running time.

For a dataset with m records and n features, in order to get coefficients securely and correctly, we use $i = 50$ iterations in the computation of inversion. Overall, we need $5mn+(3i+3)n^2+n+3i$ encryptions and $mn+(i+1)n^2+n+i$ decryptions. We also have two matrix multiplications and 3 matrix additions between encryption and decryption. Each encryption in an Azure VM takes about 0.005 seconds for each core. It is then easy to see that the encryption phase is the bottle-neck of the pre-processing phase and easily parallelizable. Since we have $5mn$ number of encryptions, by multiplying this number to the runtime of a single encryption time divided by number of cores, a good estimate of the pre-processing phase is achievable.

The estimated running time for the offline phase is given in Table 2. These estimated results are a huge improvement when compared to the previous result [19] which took two days given a dataset with only about 50,000 records and are comparable to the total running time presented in [26] in the case of 256 cores.

11 Extending the Protocol to Multiple Parties

It is easy to generalize the previous protocol to the case in which the design matrix $\tilde{X}$ and the response vector $\tilde{y}$ are shared among more than two parties.

Let A, B and C be random matrices such that $C = AB$ and let the party P_i have shares A_i, B_i, C_i of these matrices. Such triples of shares will be called matrix multiplication triples. Given a pre-distributed matrix multiplication triple and two shared matrices X and Y, it is possible to compute shares of $Z = XY$ without leaking any information about these values. The idea is for the parties to locally compute shares of $D = X - A$ and $E = Y - B$ and then open the

Table 1: Actual Time Required (in seconds) for Online Phase of Secure Protocol to Build a Predictive Linear Regression Model

Dataset Name	Number of Rows	Number of Columns	Train Time: Data Shared in the Clear	Train Time: Using Proposed Secure Protocol
Student Performance	395	30	0.3 sec	11.7 sec
Auto MPG	398	7	0.09 sec	1.2 sec
Communities and Crime	1,993	122	9 sec	147 sec
Wine Quality	4,897	11	0.9 sec	5.2 sec
Bike Sharing	17,379	13	3.7 sec	16.5 sec
State Inpatient Database (WA)	25,180	36	21 sec	93 sec
BlogFeedback	52,397	223	1,800 sec	9,000 sec
Relative Location of CT Slices on Axial Axis	53,500	367	6,000 sec	30,000 sec
YearPredictionMSD	515,344	90	3,800 sec	18,000 sec
Gas Sensor Array Under Dynamic Gas Mixtures	4,208,261	16	1,100 sec	4,500 sec

Table 2: Estimated Time Required (in seconds) for Offline Phase of Secure Protocol to Build a Predictive Linear Regression Model

Dataset Name	Number of Rows	Number of Columns	Offline Time With 16 Cores	Offline Time With 64 Cores	Offline Time With 256 Cores
Student Performance	395	30	20 sec	6 sec	2 sec
Auto MPG	398	7	4 sec	1 sec	0.3 sec
Communities and Crime	1,993	122	400 sec	100 sec	30 sec
Wine Quality	4,897	11	100 sec	30 sec	10 sec
Bike Sharing	17,379	13	350 sec	100 sec	30 sec
State Inpatient Database (WA)	25,180	36	1,500 sec	400 sec	100 sec
BlogFeedback	52,397	223	15,000 sec	4,000 sec	1,000 sec
Relative Location of CT Slices on Axial Axis	53,500	367	30,000 sec	10,000 sec	3,000 sec
YearPredictionMSD	515,344	90	70,000 sec	20,000 sec	6,000 sec
Gas Sensor Array Under Dynamic Gas Mixtures	4,208,261	16	100,000 sec	30,000 sec	10,000 sec

values D and E. The parties can compute the shares of Z using the fact that

$$Z = AE + BD + AB + DE = AE + BD + C + DE.$$

For instance, P_1 can output $Z_1 = A_1E + B_1D + C_1 + DE$ and all the other parties $Z_i = A_iE + B_iD + C_i$. The correctness can trivially be verified by inspection. The security of this method follows from the fact that, when D and E are opened, the values of X and Y are masked by the random factors A and B from the matrix multiplication triple (and all other operations are local). For the truncation protocol, one of the parties, lets say P_1, learns the value to be truncated w blinded by a factor r, the other parties only use their shares of w and r' (which represents the least significant bits of r) in order to get their outputs. The remaining protocols can be trivially generalized to the case of multiple parties.

12 Securing Against Malicious Adversaries

One possibility for obtaining security against malicious adversaries is to use the compute with MACs approach [6, 11, 10] to protect the shared values from being manipulated by a malicious adversary. In this technique there is an unconditionally secure message authentication code (MAC) associated to each shared secret value. At the beginning of the protocol the parties commit to their inputs by opening the difference between their inputs and some random shared value (that has an associated MAC). Whenever an operation is performed over some shared values, the MAC for the output value is also computed (using the MACs of the input values). This approach prevents the malicious adversary from successfully deviating from the specified protocol instructions (if he deviates, the honest parties will notice it during the checking of the MACs). The only problem for applying this technique is that the truncation protocol used before is probabilistic and so it would not allow propagation of the MACs, but this can be solved by using a deterministic truncation procedure [1] (at the cost of having to perform one execution of a comparison protocol per truncation). More details about security against malicious adversaries will be presented in the full version. See [6, 11, 10] for further details about the compute with MACs technique.

13 Conclusion

In this paper we have presented an information-theoretically secure protocol for privacy preserving linear regression in the commodity-based model. The protocol has an offline phase where a trusted authority pre-distributes random correlated data to Alice and Bob and never again engages in the protocol and/or the online phase. The online phase is orders of magnitude faster when compared to previous solutions in the literature [26, 19].

When a trusted authority is not desirable, we propose an offline phase that is computationally secure and based on any additive homomorphic public key cryptosystem. This offline phase is computationally heavy but completely parallelizable, making it practical when a large number of cores is available.

14 References

[1] M. Aliasgari, M. Blanton, Y. Zhang, and A. Steele. Secure computation on floating point numbers. In *ISOC Network and Distributed System Security Symposium – NDSS 2013*, San Diego, California, USA, Feb. 24–27, 2013. The Internet Society.

[2] D. Beaver. Precomputing oblivious transfer. In D. Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109, Santa Barbara, CA, USA, Aug. 27–31, 1995. Springer, Berlin, Germany.

[3] D. Beaver. Commodity-based cryptography (extended abstract). In *29th Annual ACM Symposium on Theory of Computing*, pages 446–455, El Paso, Texas, USA, May 4–6, 1997. ACM Press.

[4] D. Beaver. One-time tables for two-party computation. In *Computing and Combinatorics*, pages 361–370. Springer, 1998.

[5] D. Beaver. Server-assisted cryptography. In *Proceedings of the 1998 workshop on New security paradigms*, NSPW '98, pages 92–106, New York, NY, USA, 1998. ACM.

[6] R. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias. Semi-homomorphic encryption and multiparty computation. In K. G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 169–188, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany.

[7] C. Blundo, B. Masucci, D. R. Stinson, and R. Wei. Constructions and bounds for unconditionally secure non-interactive commitment schemes. *Des. Codes Cryptography*, 26(1-3):97–110, June 2002.

[8] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145, Las Vegas, Nevada, USA, Oct. 14–17, 2001. IEEE Computer Society Press.

[9] O. Catrina and A. Saxena. Secure computation with fixed-point numbers. In R. Sion, editor, *FC 2010: 14th International Conference on Financial Cryptography and Data Security*, volume 6052 of *Lecture Notes in Computer Science*, pages 35–50, Tenerife, Canary Islands, Spain, Jan. 25–28, 2010. Springer, Berlin, Germany.

[10] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart. Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In J. Crampton, S. Jajodia, and K. Mayes, editors, *ESORICS 2013: 18th European Symposium on Research in Computer Security*, volume 8134 of *Lecture Notes in Computer Science*, pages 1–18, Egham, UK, Sept. 9–13, 2013. Springer, Berlin, Germany.

[11] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 643–662, Santa Barbara, CA, USA, Aug. 19–23, 2012. Springer, Berlin, Germany.

[12] R. Dowsley, J. Graaf, D. Marques, and A. C. A. Nascimento. A two-party protocol with trusted initializer for computing the inner product. In Y. Chung and M. Yung, editors, *WISA 10: 11th International Workshop on Information Security Applications*, volume 6513 of *Lecture Notes in Computer Science*, pages 337–350, Jeju Island, Korea, Aug. 24–26, 2010. Springer, Berlin, Germany.

[13] R. Dowsley, J. Müller-Quade, A. Otsuka, G. Hanaoka, H. Imai, and A. C. A. Nascimento. Universally composable and statistically secure verifiable secret sharing scheme based on pre-distributed data. *IEICE Transactions*, 94-A(2):725–734, 2011.

[14] W. Du, Y. S. Han, and S. Chen. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In *In Proceedings of the 4th SIAM International Conference on Data Mining*, pages 222–233, 2004.

[15] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. Strauss, and R. N. Wright. Secure multiparty computation of approximations. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, pages 927–938, 2001.

[16] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. J. Strauss, and R. N. Wright. Secure multiparty computation of approximations. *ACM Transactions on Algorithms*, 2(3):435–472, 2006.

[17] O. Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004.

[18] C.-H. Guo and N. J. Higham. A schur-newton method for the matrix p'th root and its inverse. *SIAM Journal On Matrix Analysis and Applications*, 28(3):788–804, oct 2006.

[19] R. Hall, S. E. Fienberg, and Y. Nardi. Secure multiple linear regression based on homomorphic encryption. *Journal of Official Statistics*, 27(4):669–691, 2011.

[20] Y. Ishai, E. Kushilevitz, S. Meldgaard, C. Orlandi, and A. Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In *Theory of Cryptography*, pages 600–620. Springer, 2013.

[21] A. F. Karr, X. Lin, A. P. Sanil, and J. P. Reiter. Secure regression on distributed databases. *Journal of Computational and Graphical Statistics*, 14(2):263–279, 2005.

[22] A. F. Karr, X. Lin, A. P. Sanil, and J. P. Reiter. Privacy-preserving analysis of vertically partitioned data using secure matrix products. *Journal of Official Statistics*, 25(1):125, 2009.

[23] E. Kiltz, G. Leander, and J. Malone-Lee. Secure computation of the mean and related statistics. In J. Kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 283–302, Cambridge, MA, USA, Feb. 10–12, 2005. Springer, Berlin, Germany.

[24] A. C. A. Nascimento, J. Müller-Quade, A. Otsuka, G. Hanaoka, and H. Imai. Unconditionally secure homomorphic pre-distributed bit commitment and secure two-party computations. In C. Boyd and W. Mao, editors, *ISC 2003: 6th International*

Conference on Information Security, volume 2851 of *Lecture Notes in Computer Science*, pages 151–164, Bristol, UK, Oct. 1–3, 2003. Springer, Berlin, Germany.

[25] A. C. A. Nascimento, J. Müller-Quade, A. Otsuka, G. Hanaoka, and H. Imai. Unconditionally non-interactive verifiable secret sharing secure against faulty majorities in the commodity based model. In M. Jakobsson, M. Yung, and J. Zhou, editors, *ACNS 04: 2nd International Conference on Applied Cryptography and Network Security*, volume 3089 of *Lecture Notes in Computer Science*, pages 355–368, Yellow Mountain, China, June 8–11, 2004. Springer, Berlin, Germany.

[26] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 334–348. IEEE, 2013.

[27] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.

[28] R. L. Rivest. Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer. Preprint available at http://people.csail.mit.edu/rivest/Rivest-commitment.pdf, 1999.

[29] A. P. Sanil, A. F. Karr, X. Lin, and J. P. Reiter. Privacy preserving regression modelling via distributed computation. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 677–682. ACM, 2004.

[30] S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[31] R. Tonicelli, A. C. Nascimento, R. Dowsley, J. Müller-Quade, H. Imai, G. Hanaoka, and A. Otsuka. Information-theoretically secure oblivious polynomial evaluation in the commodity-based model. *International Journal of Information Security*, pages 1–12, 2014.

APPENDIX

A Problems with the Truncation Protocol of Hall et al. [19]

The protocol of Hall et al. [19] uses operations in a finite field and homomorphic encryption as a building block. A fixed-point data type is defined and also its representation in the finite field. In such approach it is necessary to perform a truncation after each multiplication, but the truncation protocol of [19] is not correct. The truncation protocol of [19] works in two phases. First, for $P \ll q$ and $\hat{x}$ such that $|\hat{x}| < P$, it takes the encoding x of $\hat{x}$ in $\mathbb{Z}_q$ (i.e., $x \in \{0, \ldots, P/2-1\} \cup \{q - P/2 + 1, \ldots, q-1\}$) and generates shares of $\hat{x}$ encoded in the smaller field $\mathbb{Z}_P$. Then using the encodings of $\hat{x}$ in both $\mathbb{Z}_q$ and $\mathbb{Z}_P$ it performs the truncation.

In their protocol Alice has the private key to an instance of Paillier's encryption scheme, Bob knows the corresponding public key q, and has the encryption under q of a value $\hat{x}$ that is encoded as $x \in \mathbb{Z}_q$, denoted $E(x)$. It is assumed that $|\hat{x}| < P$ for $P \ll q$. The parties want to truncate the input that have an additional factor M, each obtaining a share of the truncated output. It works as follows:

1. Bob draws r uniformly at random from the set $\{P, \ldots, q-1\}$. He then sets $x_{\mathsf{Bob}} = r \mod P$, computes $E(x-r)$ using the homomorphic properties of the Paillier's encryption scheme and sends it to Alice.
2. Alice decrypts the value to obtain $s = x - r \mod q$ and sets $x_{\mathsf{Alice}} = (s - q) \mod P$. She computes $E(x_{\mathsf{Alice}})$ and sends the ciphertext to Bob.
3. Bob uses the homomorphic properties of the Paillier's encryption scheme to compute $E((x_{\mathsf{Alice}} + x_{\mathsf{Bob}} - x)M^{-1} - r')$ for a random $r' \in \mathbb{Z}_q$, sends it to Alice and outputs $\left\lfloor \frac{x_{\mathsf{Bob}}}{M} \right\rfloor - r' \mod q$.
4. Alice decrypts $s' = (x_{\mathsf{Alice}} + x_{\mathsf{Bob}} - x)M^{-1} - r'$ and outputs $\left\lfloor \frac{x_{\mathsf{Alice}}}{M} \right\rfloor - s' \mod q$.

This protocol is not correct since the subprotocol for generating the shares of the secret $\hat{x}$ encoded in $\mathbb{Z}_P$ is not correct. If the blinding factor r is such that $P < x < r$ then equation 5 used in the correctness proof (see [19]) does not hold since the equality $s + r = x + q$ (in the integers) does not hold. This can be solved by adding $P/2$ to x before executing the P-sharing subprotocol and then removing $P/2$ accordingly in the end of the subprotocol.

Chapter 7

CONCLUSION

In this thesis we have proposed solutions and tools to improve quality and cost management in healthcare. To address patient data privacy concerns in the healthcare domain we have also proposed a privacy-preserving protocol for the training of a popular machine learning model, namely a linear regression model. In the cases of our machine learning solutions we were able to show improvement over previous works as well as current practices within the domain, while our linear regression protocol was shown to be an improvement over previous works with respect to security, speed, and accuracy.

Obtaining clinical and claims data from patients turned out to be a major practical challenge throughout all of our research. Building and evaluating supervised machine learning models that make predictions at the level of individual patients requires the availability of longitudinal patient data for training and testing. Because of privacy concerns however, patient IDs in records are often anonymized in such a way that it is no longer clear which records belong to the same patient. This seriously limits the options for predictive modeling, as we experienced in Chapter 3 where we were forced to formulate the "future healthcare cost prediction problem" on a quarter-by-quarter basis because we were unable to track patients across years in the available data.

Additionally, in our work on simultaneous cost and risk prediction in Chapter 5 we outlined the importance of being able to predict both risk of readmission and future costs for the same set of patients, as well as the impact machine learning techniques can have on the accuracy of such predictions. In practice, the ability to fuse claims data (often used in cost prediction) with clinical data (often used for risk prediction) for the same set of patients is hampered by the fact that this data is owned by different parties who are not in a position

to freely share private medical data.

The emerging field of secure machine learning holds promise to provide solutions to the scenarios described in the two paragraphs above. We contributed to this field in this thesis by proposing a solution to train a linear regression model when data may be held by multiple parties, as is often the case in the healthcare domain with claims data being held by insurance providers and clinical data being held by healthcare providers. With the speed and security guarantees of our protocol, our solution becomes usable in real world scenarios. It is our hope that this may lead to improved results when predicting both cost and readmission risk.

Our work was really impacted by and in response to the growth of data within the healthcare system in the United States. We have shown how machine learning can help predict costs and readmission risk and how privacy-preserving protocols can help researchers while protecting patients. Further work, however, is needed to expand what is possible for the domain in this new age of big data in healthcare. On one side, research should be done on how bringing together claims and clinical data impacts predictive model accuracy for the same set of patients. On the other hand, efficient and provably secure protocols are needed for more machine learning techniques to give the American healthcare system as many tools as possible to continue to improve the lives and health of patients.

BIBLIOGRAPHY

[1] Key features of the affordable care act. http://www.hhs.gov/healthcare/facts/timeline. Accessed on June 25, 2014.

[2] Senjuti Basu Roy, Ankur Teredesai, Kiyana Zolfaghar, Rui Liu, David Hazel, Stacey Newman, and Albert Marinez. Dynamic hierarchical classification for patient risk-of-readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1691–1700. ACM, 2015.

[3] Martine De Cock, Rafael Dowsley, Anderson CA Nascimento, and Stacey C Newman. Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2015.

[4] Stephen F. Jencks, Mark V. Williams, and Eric A. Coleman. Rehospitalizations among patients in the Medicare fee-for-service program. *New England Journal of Medicine*, 360(14):1418–1428, 2009.

[5] James Marquardt, Stacey Newman, Deepa Hattarki, Rajagopalan Srinivasan, Shanu Sushmita, Prabhu Ram, Viren Prasad, David Hazel, Archana Ramesh, Martine De Cock, and Ankur Teredesai. Healthscope: An interactive distributed data mining framework for scalable prediction of healthcare costs. In *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, pages 1227–1230. IEEE, 2014.

[6] Shanu Sushmita, Garima Khulbe, Aftab Hasan, Stacey Newman, Padmashree Ravindra, Senjuti Basu Roy, Martine De Cock, and Ankur Teredesai. Predicting 30-day risk and cost of "all-cause" hospital readmissions. In *Proceedings of the Thirtieth AAAI Workshops on Expanding the Boundaries of Health Informatics using AI*, page to appear, 2016.

[7] Shanu Sushmita, Stacey Newman, James Marquardt, Prabhu Ram, Viren Prasad, Martine De Cock, and Ankur Teredesai. Population cost prediction on public healthcare datasets. In *Proceedings of the 5th International Conference on Digital Health 2015*, pages 87–94. ACM, 2015.

[8] Steven H. Woolf and Laudan Aron, editors. *U.S. health in international perspective: Shorter lives, poorer health*. National Academies Press (US), 2013.