A Sequentialization of Features Approach to Complex Event Sequence Prediction

Chun Pan Hon

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

MASTER OF SCIENCE IN COMP SCIENCE & SYSTEMS

University of Washington

2017

Reading Committee:

Martine De Cock, Chair

Ankur Teredesai

Anderson C. A. Nascimento

Shanu Sushmita

Program Authorized to Offer Degree:
Institute of Technology

University of Washington

Abstract

A Sequentialization of Features Approach to
Complex Event Sequence Prediction

Chun Pan Hon

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

Sequence based prediction takes an ordered list of events as input and makes predictions about the next event. Most existing work on sequence based prediction assumes that the sequences are simple, i.e. consisting of symbols drawn from a small alphabet (like a DNA sequence), or consisting of numbers (like a time series). In some applications, the events are a lot more complex. In medical applications for instance, data often comes in the form of a longitudinal sequence of patient records, each of which internally contains hundreds of features of various data types. Most existing work on making predictions about the next event in complex event sequences is *event based*, meaning that only the most recent event in the sequence is used to make a prediction about the upcoming event. In this thesis we propose a new technique for *sequence based* prediction that is domain independent and that takes the order of occurrence of events into account when making predictions. The key idea is to dissect each sequence of k feature vectors of size m into a set of m simple sequences of length k, train $m \times k$ models using well established machine learning techniques such as decision trees or support vector machines, and group the $m \times k$ trained models into an ensemble for making the final prediction. We evaluate the predictive ability of our new technique by measuring the AUC for predicting risk of 30-day readmission, cost and length of stay using hospital discharge records of hundreds of thousands of congestive heart failure patients. Our experiments show that a combination of our sequence based method with an event based method gives better results than each of these methods by themselves.

TABLE OF CONTENTS

Page

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like to express my profound gratitude to Professor Martine De Cock. She has been my beacon from day one in this school. Her sincerity and mindfulness were fueling my study and research throughout the course of this degree. Her guidance and support were monumental for my work in this thesis. I am forever in debt to her.

I would like to thank the members of my advisory committee for their efforts in helping me in this research. This thesis would not be possible without their generous support.

I would also like to express immense appreciation to my parents, Chong Hon and Yuen Kiu Wan, for their encouragement and understanding for me to pursue this degree.

Finally, I owe my deepest gratitude to my wife, Cindy Fok, for everything would not make any sense without her.

DEDICATION

to my dear wife, Cindy

Chapter 1

INTRODUCTION

1.1 Background

Sequence based classification has a broad range of real-world applications. In genomic research, classifying protein sequences into existing categories is used to learn the functions of a new protein [5]. In health-informatics, classifying ECG time series (the time series of heart rates) tells if the data comes from a healthy person or comes from a patient with heart disease [23]. In anomaly detection/intrusion detection, the sequence of a user's system access activities on Unix is monitored to detect abnormal behaviors [13]. Other interesting examples include classifying query log sequences to distinguish web-robots from human users [20, 6] and classifying transaction sequence data in a bank for the purpose of combating money laundering [14].

In general, a sequence is an ordered list of events. Depending on the nature (data type) of the events, different categories of sequences can be distinguished [24]. A simple symbolic sequence is an ordered list of symbols drawn from an alphabet, like the DNA sequence $ACCCCCGT$. A complex symbolic sequence is an ordered list of sets of elements drawn from the same alphabet, like a sequence $\langle$\{milk, bread\},\{milk, egg\}, ..., \{potatos, cheese, coke\}$\rangle$ of items bought by a customer during a year. A simple time series is a list of numbers in time-stamp ascending order, like the daily closing value of the Dow Jones. A multivariate time series is an extension of this in which multiple numerical values are recorded per timestamp, e.g. consumption and income. Finally, the most challenging kind of sequences, and the least explored one so far, are complex event sequences in which events can be of arbitrarily complex data types. A prime example of this are sequences of hospital visits by the same patient, in which each visit (event) is described by a mix of numerical measurements, categorical fields and text descriptions. It is the latter kind of sequences that we are concerned with in this thesis.

Throughout this thesis we represent a sequence s as an ordered list of vectors

$$\begin{aligned} s \;=\; & \langle [x_1^{(1)}, x_2^{(1)}, \ldots, x_m^{(1)}], \\ & [x_1^{(2)}, x_2^{(2)}, \ldots, x_m^{(2)}], \\ & \ldots, \\ & [x_1^{(|s|)}, x_2^{(|s|)}, \ldots, x_m^{(|s|)}] \rangle \end{aligned} \tag{1.1}$$

We assume that each event is represented as a fixed length vector of size m. The attributes $x_1, x_2, \ldots, x_m$ that make up an event can be of different data types, including numerical and categorical data types. Not all sequences under consideration have to be of the same length. That is why in Formula (1.1), we use $|s|$ to denote the length of s, i.e. the total number of events in s. The superscript "(j)", $j = 1 \ldots |s|$, is used to denote the index of the event in the sequence.

Various interesting supervised machine learning tasks can be formulated for sequences. One popular task is to predict the correct class label for a given sequence s. For example, given a time series of ECG data for a patient, infer whether the patient is healthy or ill, or, given a DNA sequence, infer whether it belongs to a gene coding area or a non-coding area. These are examples of *sequence classification*. The task that we are concerned with in this thesis is *next event prediction*, namely, given a sequence s, predict one or more variables of the upcoming event $[x_1^{(|s|+1)}, x_2^{(|s|+1)}, \ldots, x_m^{(|s|+1)}]$.

Most existing work on making predictions about the next event in a sequence is either *event based (Markovian)* or limited to *simple symbolic sequences* [24]. *Event based* methods use only the most recent event in the sequence to make predictions about the upcoming event. Sometimes efforts are made to aggregate information of the previous events into the feature vector of the most recent event. However these aggregations are domain dependent and sacrifice the temporal dimension in the sequence. *Simple symbolic sequence* classification captures the temporal trends in the sequence and takes that into account when making predictions. However this technique only considers sequences of a single feature, with values drawn from a particular alphabet. It lacks the ability to make predictions for complex events which may contain hundreds of features of various data types.

In this thesis, we propose *a new sequence based technique for next event prediction* that

is domain independent, respects the order of occurrence of events, and considers multiple features when making predictions. This technique dissects each sequence[1] of k feature vectors of size m into a set of m simple sequences of length k, trains $m \times k$ models using well established machine learning techniques such as decision trees or support vector machines, and groups the $m \times k$ trained models into an ensemble for making the final prediction.

1.2 Motivation

The motivation of this study stems from our research in healthcare analytics. Throughout the thesis, we focus on solving 3 problems, namely predicting risk of 30-day readmission, cost and length of hospital stay, by using hospital discharge records of Congestive Heart Failure (CHF) patients. Hospital discharge records are inherently in the form of complex event sequences. Therefore, it leads us to explore different techniques to predict the next event in complex event sequences.

The 3 problems above are important due to the fact that there are about 34 million hospital admissions annually in the United States.[2] One in five patients is readmitted to the hospital within 30 days of being discharged. Many of these readmissions could be avoided by proper interventions. 30-day readmission, cost, and length of stay are commonly understood as healthcare quality measures and cost drivers in the United States [10, 8]. The ability to predict their values provides many benefits for accountable care, now a global issue and foundation for the U.S. government mandate under the Affordable Care Act. Congestive Heart Failure (CHF) is one of the leading causes of hospitalization, especially for adults older than 65 years of age [1, 17]. The prevalence and incidence of CHF have considerably increased over the past few years [17]. Studies show that CHF is one of the primary reasons behind multiple hospitalizations within a short time-span [12, 4, 22]. As we show in Chapter 4 and 5, the sequence based technique that we propose in this thesis allows to predict risk of 30-day readmission, cost, and length of hospital stay of these CHF patients more accurately.

---

[1] k is the number of events in the sequence. m is the number of attributes in a single event.

[2] `http://www.aha.org/research/rc/stat-studies/fast-facts.shtml`

1.3 Related work

A survey of sequence classification techniques was conducted by Xing *et al* [24]. The survey explores techniques for sequence classification with event sequences of single atomic features, like strings of symbols drawn from an alphabet. Process mining techniques have been developed to discover processes, i.e. transitions of activities, from within a sequence of event logs [21]. However these techniques are limited to simple sequences and are not readily extensible to large numbers of features that are typically encountered in complex events such as patient encounters.

Existing work on next event prediction for patient encounters has focused primarily on event based methods that use only the most recent event in the sequence to make predictions about the next event. Risk and cost of 30-day *all-cause* readmission has for instance been tackled by using event based machine learning techniques [19]. Data extraction and preprocessing techniques have been explored to improve prediction outcomes of risk of readmission *for CHF patients* [15]. Existing works on prediction of risk of readmission *for CHF patients* by event based classification methods [2] and hierarchical logistic regression [11] are also in place.

The research that we propose in this thesis is different from all of the above. To the best of our knowledge, the "sequentialization of features" approach that we propose in this thesis for next event prediction is entirely novel. In addition, unlike existing sequence based prediction technique, our algorithm is applicable to complex event sequences, a kind of sequences that has received little to no attention in the literature so far. This makes our technique applicable to, among other things, next event prediction over sequences of patient encounters. Our hypothesis is that this will allow us to make more accurate predictions than existing event based (Markovian) techniques in the healthcare analytics domain.

1.4 Thesis Layout

We will evaluate the predictive ability of our technique by measuring its AUC for predicting risk of 30-day readmission, cost, and length of hospital stay, using hospital discharge records of Congestive Heart Failure (CHF) patients from the California Office of Statewide Health

and Planning Development (OSHPD). We will first describe the OSHPD dataset and all the necessary steps to preprocess the data in Chapter 2. In Chapter 3 we describe a conventional event-based approach for next event prediction, and we evaluate its predictive performance for each of the 3 classification tasks mentioned above. Chapter 4 describes our proposed "sequentialization of features" approach, its evaluation with OSHPD data, and a comparison with the results from Chapter 3. As the final step, in Chapter 5, we combine the event-based approach from Chapter 3 with the sequence based approach from Chapter 4 and show through an evaluation with OSHPD data that the combination leads to better results than each of the individual methods by themselves. Finally, we conclude our work in this thesis and point out directions of future work in Chapter 6.

Chapter 2

DATA

The algorithm for next event prediction that we propose in this thesis is domain independent, and in particular independent of any specific dataset. For the ease of explaining our algorithms in Chapter 3 to 5 however, it is helpful to keep a specific application in mind. This example application is inspired by sequences of hospital admission records from inpatient data managed by the California Office of Statewide Health Planning and Development (OSHPD), the dataset that we will use to evaluate our algorithm. In this chapter, we provide details about this dataset and the preprocessing steps that we applied to it to prapare the data for consumption by the prediction algorithms presented in the next chapters.

2.1 The OSHPD dataset

The Office of Statewide Health Planning and Development[1] (OSHPD) is an organization for collecting data and disseminating information about California's healthcare infrastructure, promoting an equitably distributed healthcare workforce, and publishing valuable information about healthcare outcomes.

The OSHPD collects and publicly discloses facility level data from more than 6,000 CDPH (The California Department of Public Health) licensed healthcare facilities—hospitals, long-term care facilities, clinics, home health agencies, and hospices. These data include financial, utilization, patient characteristics, and services information. In addition, approximately 450 hospitals report demographic and utilization data on approximately 16 million inpatient, emergency department, ambulatory surgery patients, and by physician, about heart surgery patients.

In this thesis, we use a copy of the Patient Discharge Data (PDD) data set from OSHPD. We will refer to this dataset as **OSHPD dataset** for the rest of this thesis. It consists of

[1] `http://www.oshpd.ca.gov/`

a record for each inpatient discharge from a California-licensed hospital. Licensed hospitals include general acute care, acute psychiatric, chemical dependency recovery, and psychiatric health facilities. For more information on the data and reporting requirements, see the California Inpatient Data Reporting Manual[2].

The structure of the OSHPD dataset is a single table, with each row corresponding to one admission record of one patient. There are totally 15,140,658 records and 7,355,726 unique patients, with admission dates from 2009 to 2013. There are 132 attributes. A brief description of the attributes is shown in Table 2.1. Full details of the schema are publicly available on the OSHPD website[3].

2.2 Data Preprocessing

The raw data from OSHPD had a couple of issues that we have to resolve before we can use it to evaluate our algorithms. They included data inconsistencies, transfer records and invalid zip codes. As explained in detail below, these issues were fixed before we moved on to feature engineering.

2.2.1 Data inconsistencies

In the OSHPD data, each patient can have multiple admission records. For the same patient, certain demographic information, such as race and gender, should remain the same throughout all his/her admissions. However, we found that some patients have more than 1 gender and/or race in their records. The total number of patients with inconsistent gender is 18,634. The number of rows for these patients is 79,786. The number of patients with inconsistent race is 372,852. The number of rows for these patients is 1,869,468. These inconsistencies were fixed by the following steps:

- For each patient, check if gender (or race) is inconsistent
  - If inconsistencies exist, find the majority gender (or race) of this patient across all the patient's records

[2] `https://www.oshpd.ca.gov/HID/MIRCal/IPManual.html`

[3] `https://www.oshpd.ca.gov/HID/Data_Request_Center/documents/DataDictionary_Nonpublic_PDD.pdf`

  - Replace the gender (or race) in all records of this patient with the majority value
  - If there is a tie, replace the gender (or race) in all records of this patient with the value of the earliest admission

The above steps were implemented in a Python script[4]. Running the script resulted in a gender adjustment in 22,260 rows and a race adjustment in 526,972 rows.

2.2.2 Merge transfer records

After sorting all records according to the patient identifier (rln) and then admission date (admtdate), we found that some patients have the following problems in their records:

- Readmit before discharge, i.e. next admission date is earlier than the current discharge date.
- Same day readmission, i.e. next admission date is on the same day of current discharge date.

We enquired OSHPD about this and received reply explaining that same day readmission or readmission before discharge indicates that patients were transferred to a different type of care (typcare) or a different facility (oshpd_id) for treatment during the stay.

We decided to merge these transfer records in a way to retain the most important information. For each group of transfer records, each column is merged according to the following rules (see Table 2.1 for an explanation of the abbreviations):

- diag_p - Take the value from the latest record in the group
- o_diag_p - A new column to store the values of diag_p from other records in the group
- odiag(1-250) - Keep all diagnosis codes from all admissions
- proc_p - Take the value from the latest record in the group
- o_proc_p - A new column to store the values of proc_p from other records in the group
- oproc(1-250) - Keep all procedure codes from all admissions
- los_adj - Number of days between the last discharge date and first admissions date
- disp - Take the value from the latest record in the group
- scrsite, scrlic, srcroute - Take the value from the earliest record in the group

[4] `https://github.com/darrenhon/oshpd/blob/master/cleanAndTransformData.py`

- oscrsite, oscrlic, osrcroute - New columns to store the values from other records in the group
- typcare - Take the value from the latest record in the group
- otypcare - A new column to store the values of typcare from other records in the group
- poa_p - Take the value from the earliest record in the group
- admtdate - Take the value from the earliest record in the group
- dschadate - Take the value from the latest record in the group
- sev_code - Take the value from the latest record in the group
- ose_code - A new column to store the values of sev_codes from other records in the group
- charge - Sum of the values of all records in the group

The above rules are implemented in the same script as in Section 2.2.1. After running the script, the number of rows becomes 14,406,870.

2.2.3 Invalid zip codes

Invalid zip codes were found in patients and facilities. There are totally 465 facilities in the OSHPD data, of which 2 have invalid zip codes. These records were fixed by looking up the facilities' names with the identifier (oshpd_id) from OSHPD's website[5], and then looking up the correct zip codes from the facilities' websites. For patients, there are 67,887 patients with invalid zip codes in at least one of their admissions. These records were fixed by replacing the invalid zip codes with the zip code of the earlier record of the same patient if it exists and is valid. If there is no previous record or the zip code of the previous record is invalid, it was replaced with the next record. However, there are 24,886 patients having no valid zip code in all of their records. These records were fixed by replacing the zip code with the majority value of the facilities' zip codes of all admissions of the patient. A Python script is written to perform the clean up[6].

---

[5] `www.oshpd.ca.gov/hid/data_request_center/documents/app_d_facility-status.xlsx`

[6] `https://github.com/darrenhon/oshpd/blob/master/fixZip.py`

2.3 Cohort Extraction

In this thesis, we focus on patients in the Congestive Heart Failure (CHF) cohort. We define the CHF cohort as the group of patients who are diagnosed with CHF in either primary or other diagnoses in at least one of their admissions. To identify patients diagnosed with CHF, the classification of Clinical Classifications Software (CCS) is used. In the OSHPD data, diagnosis and procedure codes are stored as ICD-9 codes. ICD-9 is the 9th revision of International Classification of Diseases, an international standard for disease classification maintained by the World Health Organization (WHO). In CCS, each ICD-9 diagnosis code is assigned to one of the CCS disease categories, where CHF is one of them. In order to extract CHF cohorts, the following steps were done.

2.3.1 Flattening diagnosis and procedure codes into CCS categories

The mappings of ICD-9 diagnosis and procedure codes into CCS categories are available on the website of HCUP User Support[7]. We downloaded these mappings and implemented a Python script to parse them, and to convert ICD-9 codes into CCS. In the OSHPD data, diagnosis and procedure codes are stored as ICD-9 in multiple columns (diag_p, odiag1-24, proc_p and oproc1-20). We decided to apply one-hot encoding (flattening) on these columns and create a new set of columns for diagnosis and procedure CCS categories. The new columns for diagnosis CCS are named as "DXCCS_[CCS code]", and those for procedure CCS are "PRCCS_[CCS code]". These are all binary columns.

To flatten the ICD-9 diagnosis and procedure columns, we implemented a Python script to do the following:

- For each row, initialize all DXCCS and PRCCS columns to 0
  - Read all the diagnosis and procedure codes of the current row
  - For each diagnosis or procedure code, convert into CCS code
  - Assign 1 to the corresponding DXCCS or PRCCS column

There are totally 283 diagnosis CCS codes and 231 procedure CCS codes. So after flattening, there are totally 514 new columns in the data.

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

2.3.2 Extracting the CHF cohort

After flattening, admissions with CHF diagnosis are easily identified. The CCS code for CHF is 108. We implemented a Python script to do the following[8]:

- For each patient, search through all his/her admissions
  - For each admission, if DXCSS_108 is 1, add the patient to the CHF cohort

After extracting the cohort, the total number of rows becomes 3,082,721 and the number of patients becomes 779,330.

2.3.3 Remove patients with unknown costs

The column "charge" is renamed to "cost". We found that 156 records have a cost of $1, which indicates that no bill was generated. In addition, 358,701 records have a cost of $0, which means that the admission was at a hospital that does not report cost to OSHPD. Since cost prediction is one of the aims of our study, we omitted all records of those patients who have at least one record with a $1 cost, and/or at least one record with a $0 cost. This brings the total number of records to 2,613,895, and the number of unique patients to 660,180.

2.3.4 Remove patients with single admission

Since we are interested in predicting the cost and length of stay of future hospitalizations, we omitted patients from the study who had only one hospital admission, bringing the total number of unique patients to 497,697 and the total number of records to 2,451,412. In the remainder of this thesis, when we refer to "the CHF cohort", we mean this set of 497,697 patients.

2.3.5 Data distribution

Table 2.2 shows the distribution of the CHF cohort across the demographic characteristics. Figure 2.1 illustrates that repeated hospital admissions are common, ranging between 2 and 256 per patient. The distribution of the "Total Charges" variable, which we will refer to as

[8] `https://github.com/darrenhon/oshpd/blob/master/extractCohort.py`

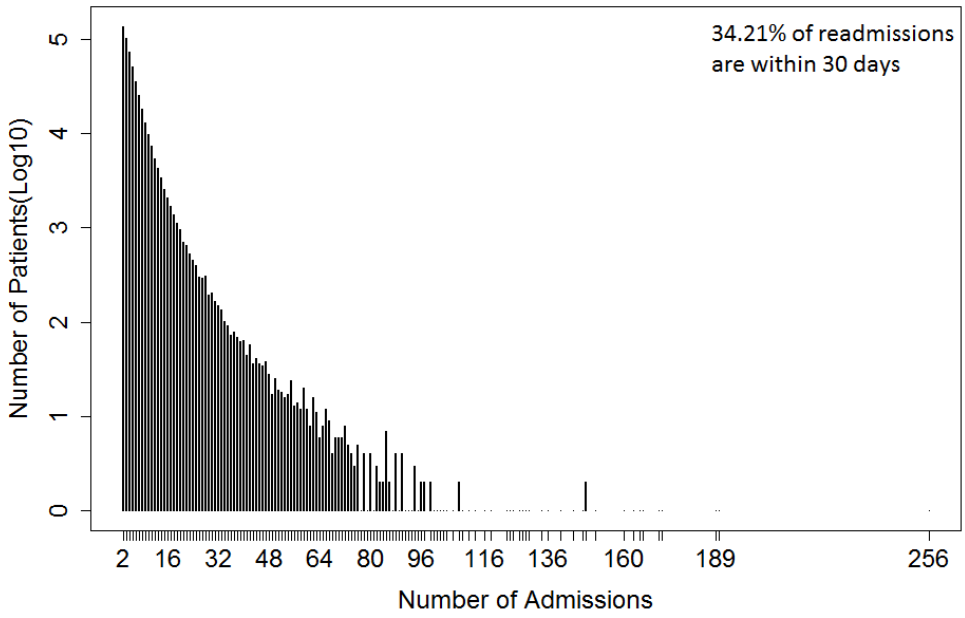


Figure 2.1: Histogram of number of admissions per patient in the CHF cohort

the cost variable from now on, exhibits a typical long tail pattern, as illustrated in Figure 2.2.

2.4 Construction of Response Variables

2.4.1 Construction of "thirtyday"

A binary column "thirtyday" is constructed to denote readmission within 30 days. The value is set to 1 if it is not the last admission of the patient and the next admission date is less than or equal to 30 days from current discharge date, and 0 if otherwise.

2.4.2 Discretization and binarization of cost and length of stay

To discretize the cost variable, we used the first quartile, median and third quartile of the CHF cohort as boundaries (see *Bucket* column of Table 2.3). To discretize the "Length of Stay" variable, we divided the range into the six buckets that are commonly used in the LACE index, namely 1, 2, 3, 4-6, 7-13, and 14+ days [25] (see *Bucket* column of Table 2.3).

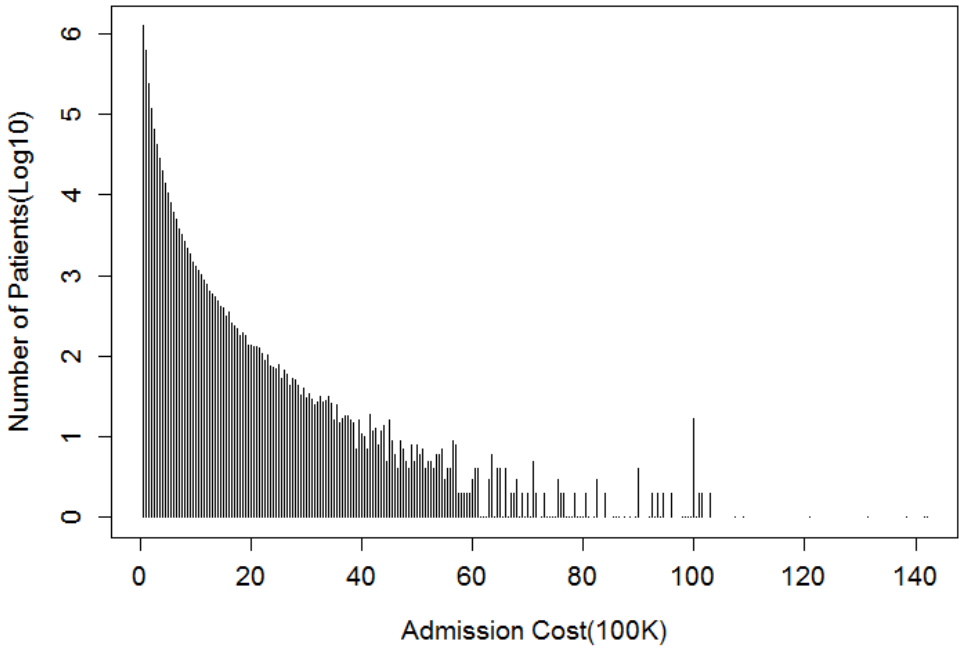


Figure 2.2: Distribution of cost, in dollars, of hospital admissions in the CHF cohort

Figure 2.3 illustrates the positive correlation between the cost and length of stay of hospital admissions in the CHF cohort. Our goal is to predict in advance which patients will go to the right top corner of the heat map, i.e. the patients with an expensive and long hospital stay. Therefore, we further binarize the cost and length of stay columns into 0 and 1 (see *Binarized Value* column of Table 2.3).

2 binary columns "nextCost_b" and "nextLOS_b" are constructed to denote the binarized cost and LOS of the next admission. Their values are 0 for the last admission of each patient.

2.5 Feature Engineering

Table 2.4 contains an overview of all the features that we use in this thesis, divided in five input feature groups and a group of response variables. Most of them are selected from recent related research [19].

Most of the features in Table 2.4 are present in the original OSHPD data or can be

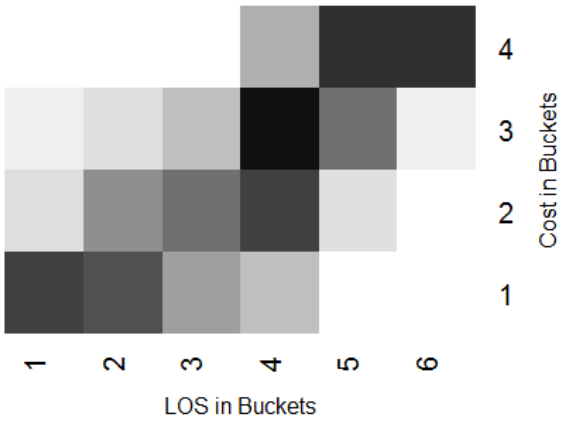


Figure 2.3: Relationship between cost and length of stay of the 2,451,412 hospital admissions in the CHF cohort. A darker color indicates the presence of more admissions in that cell of the heat map

derived in an obvious way. A Python script[9] is written to construct new features as described below.

2.5.1 Construction of "Same Day Discharge"

"Same Day Discharge" is a boolean variable that indicates whether the admission and discharge dates are the same.

2.5.2 Construction of Charlson comorbidities

Charlson comorbidities columns are binary columns that are constructed by converting all non-primary diagnosis codes of the current admission into comorbidities using the mapping defined by Quan et al [18].

2.5.3 Construction of cumulative features

The number of Emergency Department (ED) visits in the past 6 months can be derived from the variable "Source of Admission - Route" in the OSHPD data. Number of Charl-

[9] `https://github.com/darrenhon/oshpd/blob/master/bcbFeatures.py`

son comorbities can be counted after the Charlson comorbities columns are constructed. Number of unique CCS diagnoses can be counted using the "DXCCS" columns constructed during the flattening in Section 2.3.1.

LACE is regularly used in hospitals [25] to predict risk of 30 day readmission. 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 ≥ 10) is then set to determine patients with "high" readmission risk [25]. The LACE score calculation is implemented in a Python script[10].

2.6 Remove the last admission of each patient

Our goal in this thesis is to predict readmission, cost and length of stay of the next admission. Since the response variables of the last admission of each patient are all inherently 0, they do not provide any useful information. Therefore, we removed the last admission of each patient from the data. The total number of rows after the removal is 1,953,715.

2.7 Split into training, validation and testing data sets

After the above process, the data is ready for evaluating the proposed algorithms in this thesis. The data is further split into 60% training, 20% validation and 20% testing data sets randomly by patients. The training set contains 1,171,002 rows and 298,618 patients. The validation set contains 391,123 rows and 99,539 patients. The testing set contains 391,590 rows and 99,540 patients. The validation set and testing set are used in Chapter 4 and 5 only. The evaluation in Chapter 3 is done with 5-fold cross-validation.

[10] `https://github.com/darrenhon/oshpd/blob/master/lace.py`

Table 2.1: Overview of attributes in the raw OSHPD data

Attribute	**Description**
rln	Record linkage number
agyradm	Age in years at admission
sex	Patient gender
ethncty	Ethnicity
race	Race
patzip	Zip code of residence
patcnty	Country of residence
race_grp	Race group normalized
diag_p	Principal diagnosis
odiag(1-24)	Other diagnoses
ecode_p	Primary external cause of injury
ecode(1-4)	Other external causes of injury
proc_p	Principal procedure
oproc(1-20)	Other procedures
epoa_p	Present on admission - primary external cause of injury
epoa1-epoa4	Present on admission - other external causes of injury
los	Length of stay
los_adj	Adjusted length of stay
admtday	Admission day of week
admtmth	Admission month
admtyr	Admission year
admtype	Type of admission (scheduled or unscheduled)
disp	Disposition
charge	Total charge
source, srcsite, srclicns, srcroute	Source of admission(home, jail, etc)
oshpd_id	Hospital identification number
typcare	Type of care
poa_p	Present on admission - principal diagnosis
proc_pdy	Days between admission and principal procedure
opoa(1-24)	Present on admission - other diagnoses
procdy(1-20)	Other procedures days
admtdate	Admission date
dschdate	Discharge date
hplzip	Hospital zip Code
qtr_adm	Admission quarter
qtr_dsch	Discharge quarter
sev_code	Severity code
cat_code	Category code
grouper	Grouper version

Table 2.2: Distribution of the 497,697 patients in the CHF cohort across different demographics. The majority of patients is white and above 65+

		Count	%
Gender	Male	240,368	48.29
	Female	257,329	51.70
Age	0-4	574	0.11
	5-14	252	0.05
	15-24	1,693	0.34
	25-44	18,223	3.66
	45-64	111,828	22.46
	65+	365,127	73.36
Race	White	310,440	62.37
	Black	45,721	9.18
	Hispanic	92,436	18.57
	Asian/Pacific Islander	42,818	8.60
	Native American/Eskimo/Aleut	1,170	0.23
	Other	5,112	1.02

Table 2.3: Discretization and binarization of cost and length of stay variables, and distribution of the 2,451,412 discharge records across each of the "buckets"

	Bucket	Binarized Value	Range	%
Cost	1	0	\$0-\$27,320	24.99%
	2	0	\$27,320-\$48,940	25.00%
	3	0	\$48,940-\$95,770	24.99%
	4	1	\$95,770+	25.00%
LOS	1	0	1	12.39%
	2	0	2	15.07%
	3	0	3	15.52%
	4	0	4-6	27.49%
	5	1	7-13	18.53%
	6	1	14+	10.98%

Table 2.4: Overview of feature groups

DEMOGRAPHICS	Age	Numerical
	Gender	Categorical
	Race	Categorical
COST AND LOS	Cost of Current Admission (Total Charges)	Numerical
	Length of Stay of Current Admission	Numerical
ADMINISTRATIVE FEATURES	Type of Admission	Categorical
	Source of Admission	Categorical
	Source of Admission - Route	Categorical
	MS-DRG Severity Code	Categorical
	Type of Care	Categorical
	Same Day Discharge	Binary
	Disposition	Categorical
CHARLSON COMORBIDITIES	Cerebrovascular disease	Binary
	Congestive Heart Failure	Binary
	Diabetes with Chronic Complications	Binary
	Chronic Pulmonary Disease	Binary
	Dementia	Binary
	AIDS and HIV	Binary
	Mild Liver Disease	Binary
	Myocardial Infraction	Binary
	Paralysis	Binary
	Peptic Ulcer Disease	Binary
	Peripheral Disease	Binary
	Renal Disease	Binary
	Rheumatologic Disease	Binary
	Moderate or Severe Liver Disease	Binary
	Metastatic Solid Tumor	Binary
	Diabetes without Chronic Complications	Binary
	Cancer, Leukemia and Lymphoma	Binary
CUMULATIVE FEATURES	# previous admissions (excl. current adm.)	Numerical
	# ED visits in past 6 months (excl. current adm.)	Numerical
	# Charlson comorbidities so far (incl. current adm.)	Numerical
	# unique CCS diagnoses so far (incl. current adm.)	Numerical
	LACE Score	Numerical
RESPONSE VARIABLES (BINARY)	30 Day Readmission	Binary
	LOS of Next Admission	Binary
	Cost of Next Admission	Binary

Chapter 3

THE EVENT BASED APPROACH

3.1 Introduction

The event-based approach on making predictions about the next event in a sequence uses only the most recent event in the sequence. The features of event-based approach come from the feature vector of the most recent event. It does not take into account the prior history of the sequence. Sometimes efforts are made to aggregate information of the previous events into the feature vector of the most recent event. However, these aggregations are domain dependent and sacrifice the temporal dimension in the sequence. The idea is illustrated in Figure 3.1

Sequence Id	Feature x_1	...	Feature x_m	Response Variable y
s_1	a	...	β	y_1
s_1	a	...	γ	y_3
s_1	b	...	δ	y_2
s_1	b	...	α	?
s_2	b	...	β	y_1
s_2	a	...	δ	y_3
s_2	b	...	β	y_3
s_2	a	...	γ	?

Event-based approach

Figure 3.1: Features used by an event-based approach

The remainder of this chapter is structured as follows. In Section 3.2 we briefly describe the well-known supervised machine learning techniques for classification that we use

throughout this thesis. Section 3.3 describes the evaluation measures that we use in this thesis. Section 3.4 presents the results of evaluating the event-based approach with OSHPD data. Section 3.5 concludes this chapter. The work in this chapter is published in the 7th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics (ACM BCB) [9]. A copy of the paper is attached at the end of the chapter.

3.2 Event-based Classification Techniques

In this chapter, 3 common event-based classification techniques are chosen.

- **Decision Trees (DT).** Decision trees are known to be robust and expressive models. The top-down algorithms for growing decision trees can naturally handle binary as well as multi-class classification problems. The leaf nodes can refer to either of the K classes concerned. In this thesis, we use an implementation of classification and regression tree algorithm (CART) in R [3].

- **Boosted Decision Trees (AdaBoost).** Boosted decision trees are ensembles of trees. They are trained using a boosting process in which each subsequent tree is built with weighted instances which were misclassified by the previous tree [7]. Like stand-alone decision trees, these ensembles of trees can naturally handle binary as well as multi-class classification problems. Classification of a new instance with a trained ensemble of trees is based on a simple majority vote of the individual trees.

- **Logistic Regression (LR).** Logistic Regression is a discriminative classifier that models the posterior probability $P(Y|X)$ of the class Y given the input features X by fitting a logistic curve to the relationship between X and Y. As such, logistic regression model outputs can be interpreted as probabilities of the occurrence of a class [16]. 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 $P(Y = c^+|X) \geq 0.5$, then we predict that the instance belongs to the positive class, and otherwise we predict the instance belongs to the negative class. For the binary classification tasks, we use R's standard *glm* function in this thesis.

3.3 Evaluation measures

In this chapter and the rest of the thesis, we are dealing with binary classification tasks. We use area under curve (AUC) to assess the predictive power of different techniques and approaches. Below is a brief description of various binary classification evaluation measures.

In binary classification, a confusion matrix (Figure 3.2) is a useful tool to analyze test results. Figure 3.3 shows different evaluation measures for binary classification tasks based on the results in a confusion matrix.

		predicted label	
		yes	no
correct label	yes	TP	FN
	no	FP	TN

TP: the number of true positive instances
TN: the number of true negative instances
FP: the number of false positive instances
FN: the number of false negative instances

Figure 3.2: The structure of a confusion matrix

Some classifiers output a probability that an instance belongs to the positive class, and require the selection of a threshold to turn the probability into a class label. For example, we might choose to label e-mails as spam if and only if the classifier says they are spam with probability at least 0.75. Depending on how we choose the threshold (e.g. 0.5, 0.6, 0.7, 0.75, etc.), the False Positive Rate (FPR) and the True Positive Rate (TPR) will vary. We can test our classifier for various choices of the threshold, and plot the corresponding (FPR,TPR) points. The resulting curve is called a Receiver Operating Characteristic (ROC) curve. Better models appear in the top left corner of the so-called ROC space. The area under the curve (AUC) is calculated as the integral of the curve. Better classifiers have a higher AUC value.

3.4 Evaluation with OSHPD data

The performance of the event-based techniques presented in Section 3.2 are evaluated with the OSHPD data described in Chapter 2 for 3 classification tasks, namely prediction of risk of 30-day readmission, cost and length of hospital stay.

True Positive Rate (TPR); Recall; Sensitivity $\frac{\text{TP}}{\text{TP} + \text{FN}}$	Precision $\frac{\text{TP}}{\text{TP} + \text{FP}}$
True Negative Rate (TNR); Specificity $\frac{\text{TN}}{\text{TN} + \text{FP}}$	F1-measure $2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$
False Positive Rate (FPR) $\frac{\text{FP}}{\text{TN} + \text{FP}}$	Accuracy $\frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$
False Negative Rate (FNR) $\frac{\text{FN}}{\text{TP} + \text{FN}}$	Misclassification rate $\frac{\text{FP} + \text{FN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$

Figure 3.3: Evaluation measures for binary classification tasks

3.4.1 Grouping of features

The features in Table 2.4 are divided into 5 groups, namely F_1, F_2, F_3, F_4 and F_5 according to Table 3.1.

3.4.2 Testing feature groups incrementally

For each of the classification tasks (30-day readmission, cost and length of stay prediction), we built models for the 3 selected algorithms in Section 3.2, with 5 combinations of feature groups. We build our classification trees by repeatedly reducing complexity parameter (cp) from 0.01 until the tree has 50 or more nodes. The complexity parameter defines how the splits are made in the decision tree, and a split is only made if it decreases the overall

Table 3.1: Overview of feature groups

Feature	**Feature Group**
Age, Gender, Race	F_1
Cost of Current Admission (Total Charges) Length of Stay of Current Admission	F_2
Type of Admission Source of Admission Source of Admission - Route MS-DRG Severity Code Type of Care Same Day Discharge Disposition	F_3
17 Charlson comorbidities	F_4
# of previous admissions # of ED visits in past 6 months # of distinct Charlson comorbidities so far # of distinct diagnoses so far	F_5

lack of fit by at least a factor of cp. For AdaBoost, we use the *adabag*[1] implementation of boosted decision trees in R with 20 rounds of boosting. Then the tree is pruned to minimize cross-validated error.All algorithms are trained and tested using R software in a 5-fold cross-validation setup. The results are shown in Table 3.2. An example of a decision tree is shown in Figure 3.4.

Table 3.2: AUC results for different combinations of feature groups

Feature Combination	**LR**	**DT**	**ADA**
Readmission within 30-Day			
F1	0.5443	0.5379	0.5443
F1+F2	0.5631	0.5734	0.5785
F1+F2+F3	0.5958	0.5929	0.5995
F1+F2+F3+F4	0.6061	0.5980	0.6086
F1+F2+F3+F4+F5	**0.6565**	0.6411	0.6461
LOS > 6 days			
F1	0.5158	0.5276	0.5290
F1+F2	0.6057	0.6100	0.6146
F1+F2+F3	0.6077	0.6128	0.6216
F1+F2+F3+F4	0.6109	0.6159	0.6240
F1+F2+F3+F4+F5	0.6150	0.6122	**0.6285**
Cost > 95.7K			
F1	0.5300	0.5526	0.5551
F1+F2	0.6174	0.6242	0.6368
F1+F2+F3	0.6160	0.6254	0.6391
F1+F2+F3+F4	0.6180	0.6263	0.6413
F1+F2+F3+F4+F5	0.6222	0.6263	**0.6473**

LR = Logistic Regression, DT = Decision Tree, and ADA = AdaBoost.

[1] `https://cran.r-project.org/web/packages/adabag/adabag.pdf`

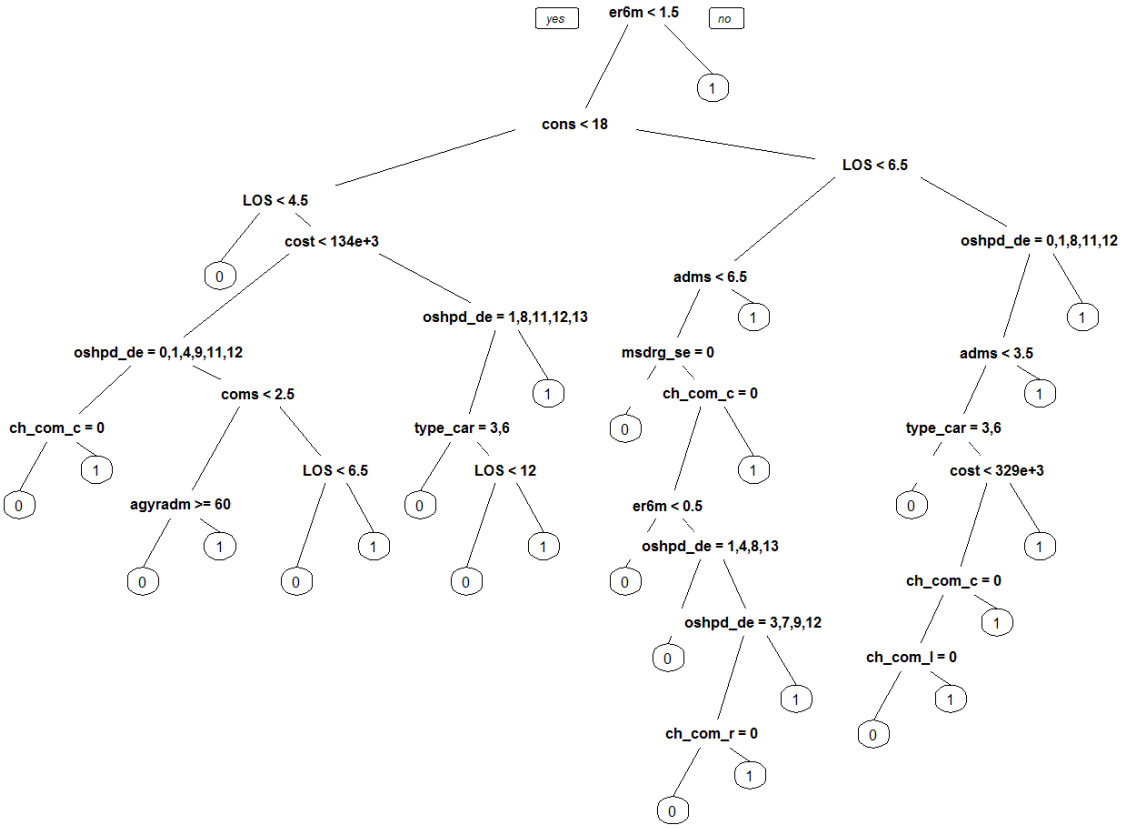


Figure 3.4: One of the decision trees for 30-day readmission prediction with all 5 groups of features

3.4.3 Observations from the results

Overall, 3 key observations can be made from the results in Table 3.2: (1) the AUC scores from all 3 methods show promise in accurately predicting early, lengthy and costly readmissions (above 60%); (2) the performance of the logistic regression models is at par with non-linear methods like decision and boosted decision trees; (3) adding more information (more features) improves the overall performance of the models. The highest AUC scores are observed when all features ($F_1 + F_2 + F_3 + F_4 + F_5$) were used. In particular, for the 30-day readmission problem, the ML models outperform the regularly used LACE index which yields an AUC of 0.6025 on the CHF OSHPD data.

3.5 Conclusion

We presented a comparative performance analysis of decision trees, boosted decision trees and logistic regression models that can flag high risk CHF patients at the time of discharge. Preliminary results show promise in using these methods for accurately stratifying patients according to 30-day readmission risk, anticipated length and cost of hospital stay.

The focus of this chapter is a pure event-based prediction, where we make a prediction for the next event (hospital admission) using only the current event, ignoring the prior history except in the cumulative features. In the coming chapters, we will introduce the proposed Sequentialization of Features approach that makes use of prior history of a sequence to predict the next event, and describe an attempt to combine both approaches.

Risk Stratification for Hospital Readmission of Heart Failure Patients: A Machine Learning Approach

Chun Pan Hon
Center for Data Science
Institute of Technology
University of Washington
Tacoma
darrencp@uw.edu

Mayana Pereira
Center for Data Science
Institute of Technology
University of Washington
Tacoma
mayanaw@gmail.com

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

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

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

ABSTRACT

Being able to stratify patients according to 30-day hospital readmission risk, anticipated length and cost of stay can guide clinicians in discharge planning and intervention recommendation, leading to an increase of quality of care, and a decrease of healthcare cost. We present a comparative performance analysis of decision trees, boosted decision trees and logistic regression models that can flag, at the time of discharge, patients with an anticipated early, lengthy and expensive readmission. We validate our models using discharge records of 500K congestive heart failure patients from California-licensed hospitals.

Categories and Subject Descriptors

D.2.8 [**Life and Medical Sciences**]: Health; I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation, Feature evaluation and selection*

1. INTRODUCTION

There are about 34 million hospital admissions annually in the U.S., with 1 in 5 patients being readmitted to the hospital within 30 days of being discharged. Congestive heart failure (CHF) is one of the leading causes of hospitalization [2]. Many CHF related readmissions could be avoided by proper interventions. While predicting 30-day readmission has been identified as one of the key problems for the healthcare domain, not many solutions are known to be effective [3]. To improve the clinical process for CHF patients, healthcare organizations still leverage the proven best-practices, called "*Get With The Guidelines*" by the American Heart Association. Furthermore, uncertainty in length of hospital stay is a major deterrent to effective scheduling for admission of elective patients. A model to predict the Length of Stay (LOS) for hospitalized patients can be an effective tool for providers, as it will enable early interventions to prevent complications, among other things [1]. However, the ability to risk stratify for LOS is limited, and more challenging for CHF patients. Readmissions and prolonged hospital stay act as substantial contributors to rising healthcare costs [2]. Existing cost prediction models are primarily focused on 'general' healthcare costs as opposed to hospital admissions, and are often rule based and regression models.

In recent years, governments have started to make healthcare data available for research. Analytics solutions that leverage this data are central to improving accountability in care, but many state-of-the-art machine learning (ML) approaches remain unexplored so far in the healthcare analytics domain. In this study, we leverage longitudinal inpatient data from the California Office of Statewide Health Planning and Development (OSHPD) to train and test ML models for predicting, at the time of discharge, *(1) whether the next admission of the patient will be within 30 days, (2) whether the hospital stay of the next admission will be long*, and *(3) whether the cost of the next admission will be high*. For each of the classification tasks under study, we build logistic regression models, decision trees and boosted decision trees. We investigate the use of different demographic and clinical features, and observe how adding more feature groups improves the performance of the models. To the best of our knowledge, our work is the first effort to build and validate ML models for risk stratification of CHF patients in the OSHPD data.

Table 1: Class distributions in the dataset

Task	Positive Class	Negative Class
30-Day	34.21%	65.78%
LOS (> 6 days)	29.82%	70.17%
Cost (> 95.7K)	25.08%	74.91%

BCB '16 October 02-05, 2016, Seattle, WA, USA

ACM ISBN 978-1-4503-4225-4/16/10.
DOI: http://dx.doi.org/10.1145/2975167.2985648[dx.doi.org]

Table 2: Overview of feature groups

Feature	Feature Group
Age, Gender, Race	F_1
Cost of Current Admission (Total Charges) Length of Stay of Current Admission	F_2
Type of Admission Source of Admission Source of Admission - Route MS-DRG Severity Code Type of Care Same Day Discharge Disposition	F_3
17 Charlson comorbidities	F_4
# of previous admissions # of ED visits in past 6 months # of distinct Charlson comorbidities so far # of distinct diagnoses so far	F_5

2. METHODS AND RESULTS

We requested non public OSHPD data for the years 2009-2013. The dataset is a collection of records in tabular format with each row corresponding to one hospital discharge record of one patient. After performing a series of data preprocessing steps (see [4] for a description of similar steps), we extracted all the records of all patients who have CHF as a primary or secondary diagnosis in at least one of their records.[1] Since we are interested in predicting the cost and length of stay of future hospitalizations, we omitted patients from the study who had only one hospital admission, bringing the total number of unique patients to 497,697 and the total number of records to 2,451,412. Table 2 shows an overview of all the features used. Some of them are taken directly from the raw OSHPD data[2], while others are constructed.

We trained and tested ML models for 3 binary classification problems at the time of discharge: next admission within 30 days, length of stay more than 6 days, and cost of next admission to be high, i.e. above $95.7K. A threshold of 30 days was chosen because the 30-day-readmission rate of patients is a commonly used quality metric in the U.S.. The threshold for length of stay stems from the discretization used in the popular LACE index [5]. Finally, $95.7K for the cost threshold is derived directly from the OSHPD data, splitting off the highest quartile. For each of the classification tasks, we built logistic regression (LR) models, decision trees (DT) and boosted decision trees (ADA). All algorithms are trained and tested using R software in a 5-fold cross-validation setup. Since the data is imbalanced (see Table 1), in each fold we undersampled the training data by randomly selecting instances from the majority class to match with the number of samples of the minority class. After undersampling both classes have the same number of instances in the training data. No undersampling was done on the test data.

Overall, 3 key observations can be made from the results in Table 3: (1) the AUC scores from all 3 methods show promise in accurately predicting early, lengthy and costly readmissions (above 60%); (2) the performance of the logistic regression models is at par with non-linear methods like decision and boosted decision trees; (3) adding more information (more features) improves the overall performance of the models. The highest AUC scores are observed when all features ($F_1 + F_2 + F_3 + F_4 + F_5$) were used. In particular, for the 30-day readmission problem, the ML models outperform the regularly used LACE index which yields an AUC of 0.6025 on the CHF OSHPD data.

[1]Using the ICD-9-CM codes for CHF: 398.91 and 428.XX.
[2]http://www.oshpd.ca.gov/HID/Data_Request_Center/documents/PDD_Nonpublic_DataDictionary.pdf

Table 3: AUC results for different combinations of feature groups from Table 2. LR = Logistic Regression, DT = Decision Tree, and ADA = AdaBoost.

Feature Combination	LR	DT	ADA
Readmission within 30-Day			
F1	0.5443	0.5379	0.5443
F1+F2	0.5631	0.5734	0.5785
F1+F2+F3	0.5958	0.5929	0.5995
F1+F2+F3+F4	0.6061	0.5980	0.6086
F1+F2+F3+F4+F5	**0.6565**	0.6411	0.6461
LOS > 6 days			
F1	0.5158	0.5276	0.5290
F1+F2	0.6057	0.6100	0.6146
F1+F2+F3	0.6077	0.6128	0.6216
F1+F2+F3+F4	0.6109	0.6159	0.6240
F1+F2+F3+F4+F5	0.6150	0.6122	**0.6285**
Cost > 95.7K			
F1	0.5300	0.5526	0.5551
F1+F2	0.6174	0.6242	0.6368
F1+F2+F3	0.6160	0.6254	0.6391
F1+F2+F3+F4	0.6180	0.6263	0.6413
F1+F2+F3+F4+F5	0.6222	0.6263	**0.6473**

3. CONCLUSION

We presented a comparative performance analysis of decision trees, boosted decision trees and logistic regression models that can flag high risk CHF patients at the time of discharge. Preliminary results show promise in using these methods for accurately stratifying patients according to 30-day readmission risk, anticipated length and cost of hospital stay. In our future research, we aim to investigate additional state-of-the-art ML methods, as well as improve our existing models through feature engineering.

4. REFERENCES

[1] P. R. Hachesu, M. Ahmadi, S. Alizadeh, and F. Sadoughi. Use of data mining techniques to determine and predict length of stay of cardiac patients. *Healthc Inform Res*, 19(2):121–129, Jun 2013.

[2] 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.

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

[4] M. Pereira, V. Singh, C. P. Hon, T. G. McKelvey, S. Sushmita, and M. De Cock. Predicting future frequent users of emergency departments in California state. In *Proceedings of the 1st Workshop on Methods and Applications for Healthcare Analytics (MAHA) in conjunction with ACM BCB 2016*, 2016.

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

Chapter 4

THE SEQUENTIALIZATION OF FEATURES APPROACH

4.1 Main Algorithm

Recall from the introduction that the problem under study in this thesis is: given a sequence s, as an ordered list of events,

$$\begin{aligned} s \;=\; & \langle [x_1^{(1)}, x_2^{(1)}, \ldots, x_m^{(1)}], \\ & [x_1^{(2)}, x_2^{(2)}, \ldots, x_m^{(2)}], \\ & \ldots, \\ & [x_1^{(|s|)}, x_2^{(|s|)}, \ldots, x_m^{(|s|)}] \rangle \end{aligned} \tag{4.1}$$

predict one or more variables of the upcoming event

$$[x_1^{(|s|+1)}, x_2^{(|s|+1)}, \ldots, x_m^{(|s|+1)}]$$

Without loss of generality, let us focus on the prediction of one particular variable x_i, henceforth called "the response variable" and let us simply denote it as y in the remainder of this thesis. Note that the task of predicting the correct value of y for a given sequence s can be conceived as a *sequence classification task* (or a *sequence regression task*, if y is of a continuous numeric data type). An example of a response variable in the healthcare analytics scenario from Chapter 2 could be the cost or the length of the next hospital admission of the patient.

Table 4.1 shows a small example of sequence data that can be used for training and testing purposes. The table contains data from two sequences s_1 and s_2, each of which have 4 consecutive events. Each event consists of values for m features. In addition, for each event we have the value of the response variable. Table 4.2 shows a very similar table but with an explicit illustration of the prediction problem that we solve in this thesis, namely predicting the final response variable in the sequence, as indicated by the question marks in Table 4.2.

Table 4.1: Sample sequence data that can be used for training and testing

Sequence Id	Feature x_1	...	Feature x_m	Response Variable y
s_1	a	...	β	y_1
s_1	a	...	γ	y_3
s_1	b	...	δ	y_2
s_1	b	...	α	y_2
s_2	b	...	β	y_1
s_2	a	...	δ	y_3
s_2	b	...	β	y_3
s_2	a	...	γ	y_2

Table 4.2: Illustration of the prediction problem

Sequence Id	Feature x_1	...	Feature x_m	Response Variable y
s_1	a	...	β	y_1
s_1	a	...	γ	y_3
s_1	b	...	δ	y_2
s_1	b	...	α	**?**
s_2	b	...	β	y_1
s_2	a	...	δ	y_3
s_2	b	...	β	y_3
s_2	a	...	γ	**?**

Table 4.3: Sequentialized table of feature x_1

X_{e1}	X_{e2}	X_{e3}	X_{e4}	Response Variable y
			a	y_1
			a	y_3
			b	y_2
			b	y_2
			b	y_1
			a	y_3
			b	y_3
			a	y_2
		a	a	y_3
		a	b	y_2
		b	b	y_2
		b	a	y_3
		a	b	y_3
		b	a	y_2
	a	a	b	y_2
	a	b	b	y_2
	b	a	b	y_3
	a	b	a	y_2
a	a	b	b	y_2
b	a	b	a	y_2

By extracting sequences of feature x_1 from Table 4.1, we have a table of feature x_1 sequences and response variable y. X_{ei} denotes the i-th event in the sequence of feature x_1.

The proposed algorithm is divided into training, validation, and testing phases. Assume the data has gone through necessary preprocessing such as cleaning and feature engineering. Split the data into training, validation and testing sets.

The training phase of the proposed algorithm is described as follows:

1. Construct "sequentialized" tables for each feature

   For the training data, construct a new, separate table for each feature x by extracting all subsequences of the column for x in the original table, and associating them with the value of their response variable from the original table. Table 4.3 illustrates the sequentialized table for x_1 constructed based on the information from Table 4.1.

2. Divide each sequentialized table into subtables

   For each sequentialized table, divide it into subtables according to the length of sequence. As illustrated by the dashed lines in Table 4.3, the sequentialized table is divided into 4 subtables of sequence length 1, 2, 3 and 4.

3. Train machine learning models for each subtable

   For each subtable, apply well-known supervised machine learning techniques to predict y, such as decision trees, logistic regression, random forests, etc. The choice of machine learning technique will depend on the data type of the selected feature and the response variable, since they can be continuous, discrete, or categorical. This process is repeated to all subtables of all sequentialized tables. Assuming there are m features and l subtables for each feature, there will be totally $m \times l$ individual models to be trained.

The steps to make prediction on new data are described as follows:

1. Sequentialize new data for each feature

   When new data comes in as a sequence of the form of Table 4.1, extract the sequence for each of the m features.

2. Make prediction for each sequentialized feature

   For each feature sequence, locate the model with the corresponding sequence length and apply the model to make a prediction.

3. Combine the predictions of each feature in an ensemble

   After using all feature sequences to make predictions, combine them in an ensemble to obtain the final prediction. Let us denote the prediction and weight of the i-th feature by p_i and w_i. The final prediction P is obtained by the formula,

$$P = p_1 w_1 + p_2 w_2 + \ldots + p_m w_m \tag{4.2}$$

   where

$$\sum_{i=1}^{m} w_i = 1 \tag{4.3}$$

The weights are determined during the validation phase. The validation phase is described as follows:

1. Unweighted ensemble

   For an unweighted ensemble, simply assign equal values of $1/m$ to all weights.

2. Weighted ensemble

   For a weighted ensemble, the weights have to be optimized with regard to AUC, accuracy, or any desired criteria. The weights can be optimized with any optimization algorithm by applying the above prediction steps to validation data repeatedly.

After getting the weights, the ensemble can be tested by applying the above prediction steps to the testing data. Figure 4.1 shows the overall workflow of our proposed algorithm.

4.2 Evaluation with OSHPD data

We evaluate the proposed approach by applying it to the OSHPD data to predict 30-day readmission, cost and length of stay. The data has gone through all the cleaning and preprocessing steps as described in Chapter 2. The same feature selection and binarization of cost and length of stay is applied as in Chapter 3. The data is divided into 60% training, 20% validation and 20% testing.

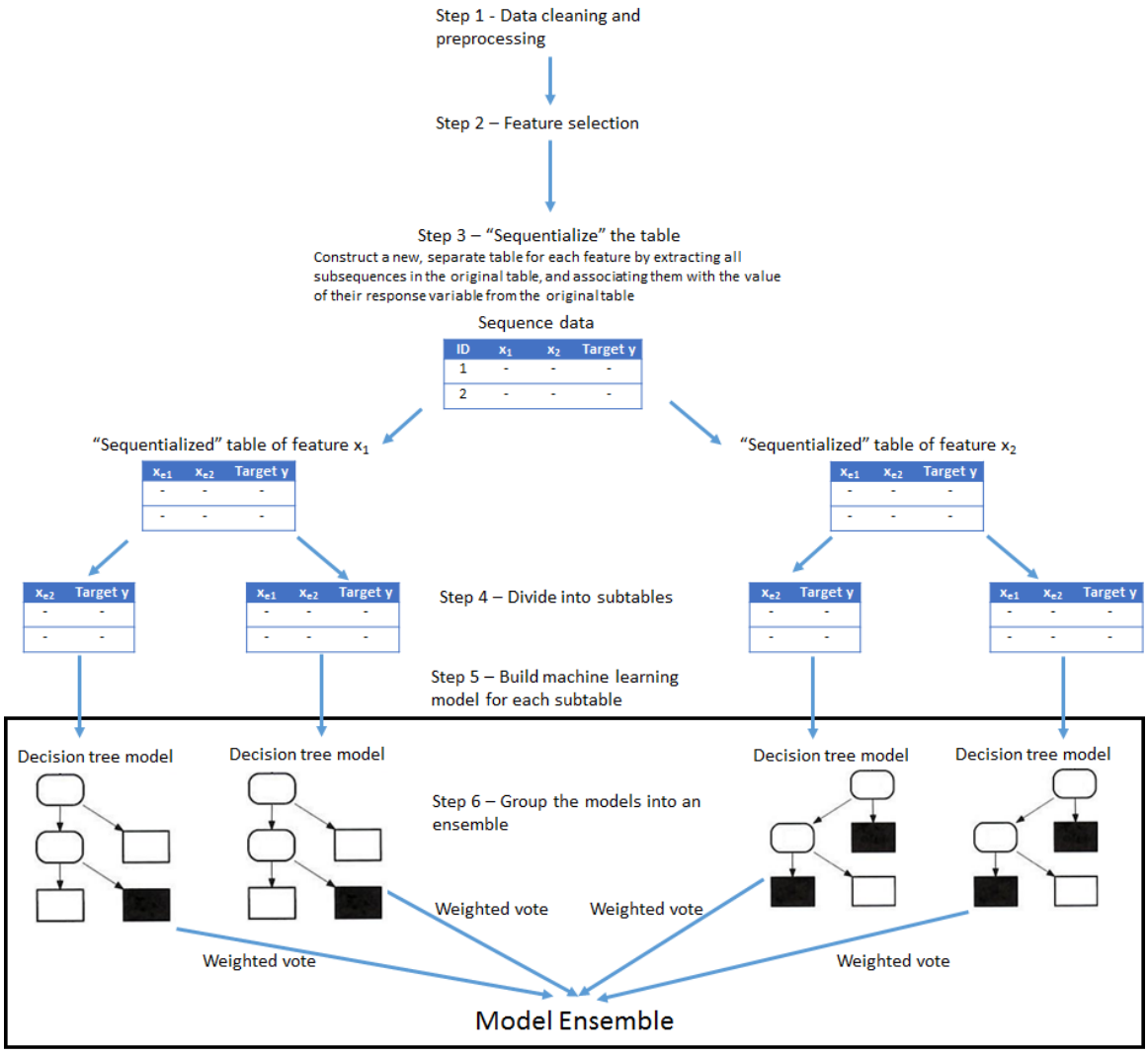


Figure 4.1: Overall workflow of our "Sequentialization of Features" algorithm

For every feature a table with extracted subsequences is created, over which a classification model is trained using a base classifier (like decision trees). The final classification is performed by an ensemble over all the base classifiers.

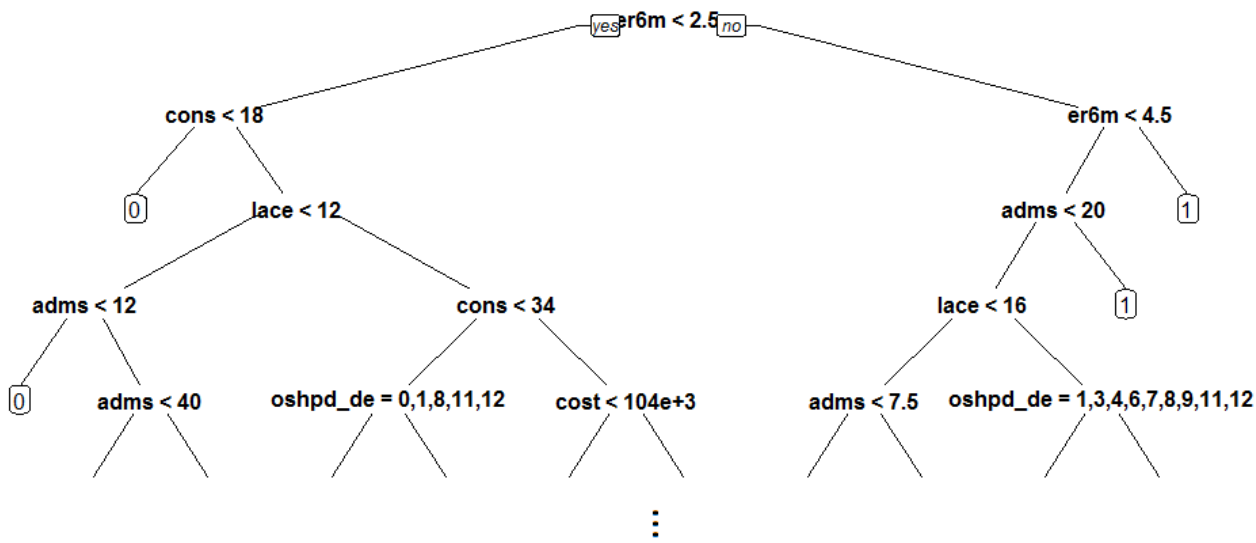


Figure 4.2: Top 5 levels of event-based decision tree for 30-day readmission.

The tree in full size has 209 nodes and is too big to show here. So only top 5 levels are shown. In this figure, *er6m* is ED visits in the past 6 months. *cons* is the number of unique CCS diagnoses. *adms* is the number of preivous admissions. *oshpd_de* is disposition. *lace* is the LACE score. Comparing with the 4 features selected by sequentialization from Table 4.4, number of ED visits in the past 6 months, disposition and LACE score are found here, but LOS is missing. And number of previous admissions, number of unique CCS diagnoses, and cost are not selected from Table 4.4. 3 out of 6 features in this figure are selected from Table 4.4. It shows that features that are important for event-based readmission prediction are similar to but not the same as those for sequence-based.

4.2.1 Baseline decision tree

An R script[1] is written to build a decision tree for each of the 3 problems (30-day readmission, cost and length of stay) in the conventional event-based way, making use of all 35 features in Table 2.4. In contrast to Chapter 3, the cp value of these trees are not optimized for proper depth. These trees will serve as a baseline for comparison with the sequentialized approach prepared in this chapter. They are shown in Figure 4.2, 4.3 and 4.4.

[1] `https://github.com/darrenhon/projects/blob/master/thesis/buildDT.r`

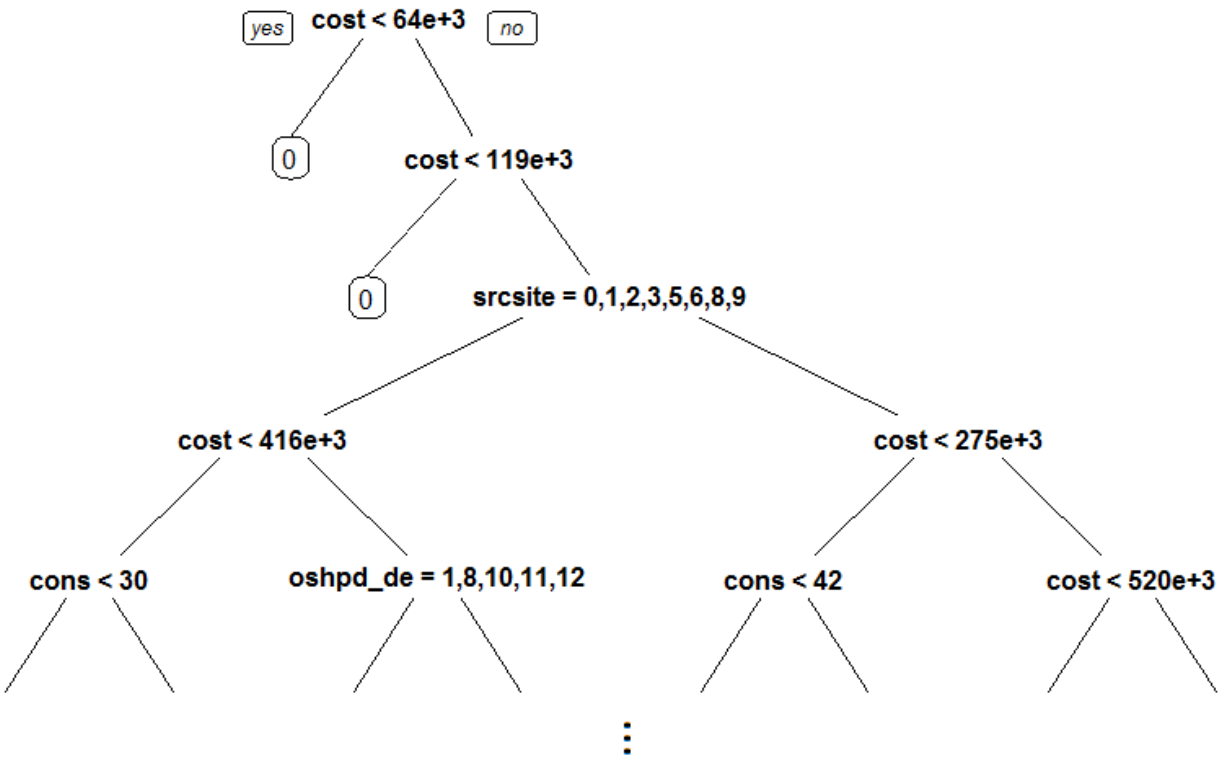


Figure 4.3: Top 5 levels of event-based decision tree for cost.

The tree in full size has 131 nodes and is too big to show here. So only top 5 levels are shown. In this figure, *srcsite* is source of admission. *cons* is the number of unique CCS diagnoses. *oshpd_de* is disposition.

Comparing with the 4 features selected by sequentialization from Table 4.4, cost and source of admission are found here, but LOS and LACE score are missing. And number of unique CCS diagnoses, and disposition are not selected from Table 4.4. 2 out of 4 features in this figure are selected from Table 4.4. It shows that features that are important for event-based cost prediction are similar to but not the same as those for sequence-based.

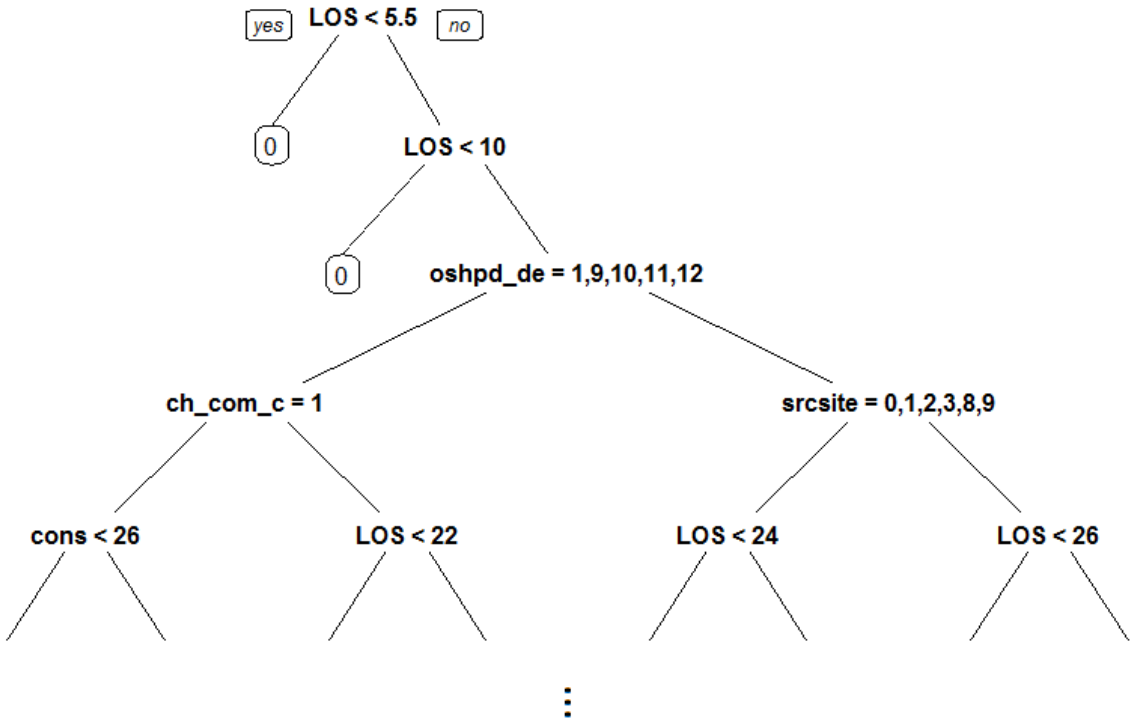


Figure 4.4: Top 5 levels of event-based decision tree for length of stay.

The tree in full size has 159 nodes and is too big to show here. So only top 5 levels are shown. In this figure, *oshpd_de* is disposition. *srcsite* is source of admission. *ch_com_c* is the Charlson comorbidity of Congestive Heart Failure (CHF). *cons* is the number of unique CCS diagnoses.

Comparing with the 4 features selected by sequentialization from Table 4.4, LOS and disposition are found here, but cost and LACE score are missing. And Charlson comorbidity of CHF, source of admission and number of unique CCS diagnoses are not selected from Table 4.4. 2 out of 5 features in this figure are selected from Table 4.4. It shows that features that are important for event-based LOS prediction are different from those for sequence-based.

4.2.2 Sequentializing the table

A Python script[2] is written to sequentialize a feature into a sequentialized table and divide it into subtables as shown in Table 4.3. The maximum length of a sequence is limited to 20. Sequences longer than 20 are truncated for the construction of the sequentialized table. A shell script[3] is written to repeat this process for all features in Table 2.4. As a result, after sequentializing all 35 features, we have 700 sequentialized subtables. This process is repeated for the 3 problems.

4.2.3 Training the decision trees models

An R script[4] is used to train decision trees on each of the sequentialized subtables. The trees are pruned to minimize cross-validated error. Due to the large number of trees that have to be trained, i.e. 700 trees for each of the 3 problems, and computation limitations, the trees have hard-coded cp value of 0.00001. This is different from Chapter 3 where trees have cp values optimized for proper tree depth. As a result, the decision trees here may not perform as good as those in Chapter 3. However, we decided this is not an issue since the purpose here is to compare with the conventional event-based trees. As long as the trees are built in the same way the comparison stays valid. For each of the 3 problems, 700 decision trees are trained and persisted on disk.

Examples of decision trees are shown in Figure 4.5 and 4.6. As we can see from these figures, the nodes in the trees are dominated by events closer to the end of the sequence. This conforms with our intuition that the next event is more related to the most recent events than the older ones.

---

[2] `https://github.com/darrenhon/projects/blob/master/thesis/flip.py`

[3] `https://github.com/darrenhon/projects/blob/master/thesis/flipall.sh`

[4] `https://github.com/darrenhon/projects/blob/master/thesis/buildModels.r`

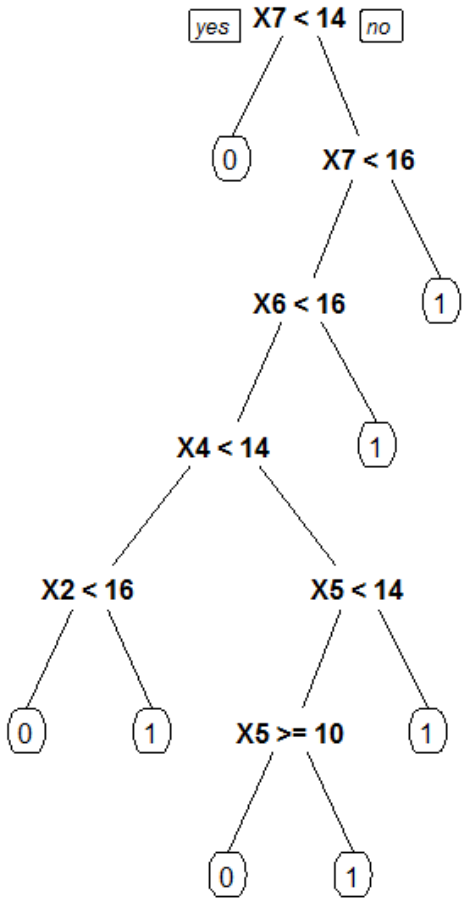


Figure 4.5: Decision tree of sequentialized subtable. The feature being sequentialized is LACE score. Sequence length is 7. Response variable is 30-day readmission.

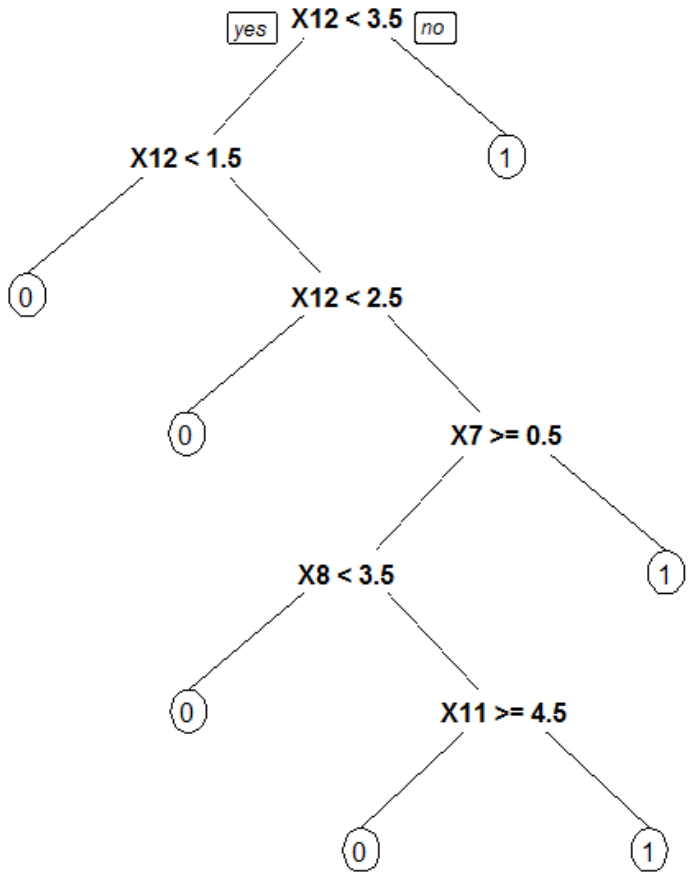


Figure 4.6: Decision tree of sequentialized subtable. The feature being sequentialized is number of ED visits in the past 6 months. Sequence length is 12. Response variable is 30-day readmission.

4.2.4 Second feature selection

An R script[5] is implemented as described in 4.1 to evaluate the sequentialized decision trees with AUC as metric. Due to computation limitations, we are unable to optimize weights for all 35 features. To further reduce the number of features, we evaluate each feature independently with validation data to obtain the AUCs. The list of AUC values are shown in Table 4.4. As seen from the table, only a few features have exceptionally high AUC values, the rest of them are almost the same. So we decided to take the 4 features with the highest AUC for each problem. The number 4 is the highest number of features that can finish the validation and testing phases within 12 hours.

4.2.5 Build baseline decision trees with selected features

After 4 features are selected for each problem, another group of event-based decision trees are built with the selected features. These trees, along with those built with 35 features in Section 4.2.1, will serve as baseline for comparison with sequentialization approaches.

4.2.6 Parameters tuning

The R script in the previous step can take a list of features and maximize the AUC by varying the weights of each feature with an optimizer function based on Nelder–Mead algorithm. For each problem, we supply the 4 selected features to the script with validation data. The optimized weights for each problem are shown in Table 4.5, 4.6, and 4.7. The optimized weights are persisted into disk for further evaluation with testing data.

4.2.7 Testing phase

5 evaluations were done with testing data for each problem. The first one is a conventional event-based approach with all 35 features, using the decision trees described in Section 4.2.1. The second one is a conventional event-based approach with 4 selected features, using the decision trees described in Section 4.2.5. The third one is unweighted sequentialization with all 35 features. There are 20 decision trees for each feature, totally 700 trees in the

[5] `https://github.com/darrenhon/projects/blob/master/thesis/runModels.r`

Table 4.4: AUC of sequentialized tables decision trees of individual features.

Feature	AUC (30-day)	AUC (Cost)	AUC (LOS)
Age	0.6076	0.5080	0.5214
Gender	0.6131	0.5100	0.5217
Race	0.6134	0.5099	0.5217
Cost of Current Admission	0.6128	**0.5902**	**0.5536**
Length of Stay of Current Admission	**0.6222**	**0.5606**	**0.5877**
Type of Admission	0.6116	0.5100	0.5217
Source of Admission	0.6063	**0.5144**	0.5271
Source of Admission - Route	0.6123	0.5100	0.5217
MS-DRG Severity Code	0.5911	0.5100	0.5217
Type of Care	0.6028	0.5100	0.5270
Same Day Discharge	0.6112	0.5100	0.5217
Disposition	**0.6172**	0.5138	**0.5503**
Cancer	0.6135	0.5100	0.5217
Cerebrovascular disease	0.6120	0.5100	0.5217
Congestive Heart Failure	0.6110	0.5100	0.5217
Diabetes with Chronic Complications	0.6093	0.5100	0.5217
Chronic Pulmonary Disease	0.6101	0.5100	0.5217
Dementia	0.6131	0.5100	0.5217
AIDS and HIV	0.6133	0.5100	0.5217
Mild Liver Disease	0.6132	0.5100	0.5212
Myocardial Infraction	0.6092	0.5100	0.5216
Paralysis	0.6132	0.5100	0.5217
Peptic Ulcer Disease	0.6102	0.5100	0.5217
Peripheral Disease	0.6115	0.5100	0.5217
Renal Disease	0.6064	0.5099	0.5217
Rheumatologic Disease	0.6133	0.5100	0.5217
Moderate or Severe Liver Disease	0.6131	0.5100	0.5217
Metastatic Solid Tumor	0.6147	0.5100	0.5217
Diabetes without Chronic Complications	0.6103	0.5100	0.5217
No. of Previous Admissions	0.6134	0.5100	0.5217
No. of ED Visits in Past 6 Months	**0.6207**	0.5100	0.5217
No. of Distinct Charlson Comorbidities	0.6067	0.5100	0.5217
No. of Distinct Diagnoses	0.6126	0.5100	0.5217
LACE Score	**0.6225**	**0.5264**	**0.5412**

The trees are trained on the training data (60%) and the AUC values are computed on the validation data (20%). The testing data (20%) was not used at all in this process.

Table 4.5: Optimized weights for 4 best features of 30-day readmission

Length of Stay	Disposition	ED Visits in 6 Months	LACE Score
0.2500	0.2500	0.2500	0.2497

Table 4.6: Optimized weights for 4 best features of Cost

Cost	Length of Stay	Source of Admission	LACE Score
0.2561	0.2392	0.2565	0.2480

Table 4.7: Optimized weights for 4 best features of LOS

Cost	Length of Stay	Disposition	LACE Score
0	0.3912	0.2180	0.3906

ensemble. The fourth one is unweighted sequentialization with 4 features. The last one is sequentialization with optimized weights on 4 features. The latter two approaches have 20 decision trees for each feature, totally 80 trees in the ensemble. The resulting AUC values are shown in Table 4.8.

4.2.8 Observations from the results

Comparing with Chapter 3, the event-based trees in this chapter have different AUC values. This is due to 2 reasons. Firstly, as mentioned in Section 4.2.1, the trees in this chapter are not optimized for proper depth. Secondly, the predictions in Chapter 3 were made at admission level, while those in this chapter were made at patient level. At admission level, we made 351,590 predictions. At patient level, we made 99,540 predictions.

As shown in Table 4.8, the unweighted sequentialization with all features has the lowest AUC. Unweighted sequentialization with 4 features have significantly higher AUC for all

3 problems. It shows that for unweighted sequentialization, reducing number of feature is very important. This is due to the fact that unweighted sequentialization gives equal weights to all features, and thus the more feature we use the more likely we are introducing noise into the model. The situation is different for weighted sequentialization because when we are optimizing the weights, the best features will get the highest weights, and the worst ones will be eliminated by getting 0 weights. So for weighted sequentialization, the more features, the better is the model.

Comparing with Table 4.4, the AUC of weighted or unweighted sequentialization with 4 features are better than any feature individually for all 3 problems. It shows that the idea of sequentialization of features approach helps improve the performance.

In the case of cost and LOS, optimizing the weights also improves the performance slightly. For 30-day readmission, the optimized weights are almost identical to average weights (0.25). Therefore it is expected to have very close AUC for weighted and unweighted approaches.

Comparing with event-based approaches, cost and LOS performed the best with weighted sequentialization, while 30-day readmission performed the best with 4 feature event-based model. This is possibly because cost and LOS exhibit a stronger sequential patterns while 30-day readmission is weaker.

4.2.9 Further analysis

Filter data by sequence length

The weighted sequentialization approach is further evaluated by repeating the above with testing data filtered into different sequence lengths. The results are shown in Figure 4.7.

It can be observed in this figure that all 3 problems have good AUC at sequence lengths between 8 to 10. 30-day exhibits a stronger sequential tendency on very long (>18) sequences. Cost and LOS wobble on longer sequence lengths and do not improve significantly on very long sequences.

Table 4.8: AUC of event-based approach with 35 features, event-based approach with 4 features, unweighted sequentialization approach with 35 features, unweighted sequentialization approach with 4 features, and weighted sequentialization approach with 4 features

	30-day	Cost	LOS
Event-based 35 features	0.6156	0.5842	0.5808
Event-based 4 features	**0.6295**	0.5835	0.5801
Unweighted Sequentialization 35 features	0.5998	0.5705	0.5658
Unweighted Sequentialization 4 features	0.6241	0.5897	0.5915
Weighted Sequentialization 4 features	0.6241	**0.5899**	**0.5979**

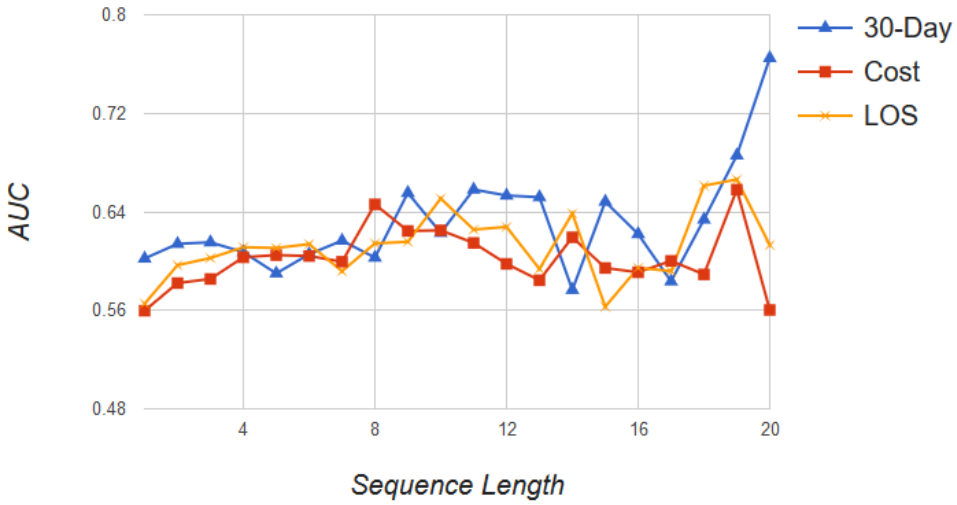


Figure 4.7: AUC for different sequence lengths of weighted sequentialization approach with 4 features

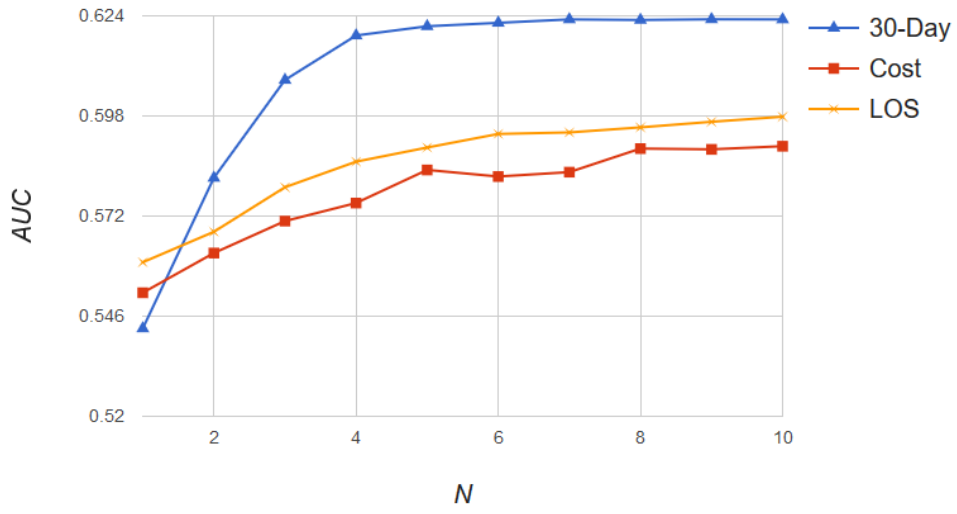


Figure 4.8: AUC for taking the last N elements of weighted sequentialization approach with 4 features

Take the last N elements of each sequence

Another analysis is done by repeating the weighted sequentialization approach with testing data while taking only the last N elements of each sequence to feed into the decision trees, effectively truncating sequences longer than N elements. The results are shown in Figure 4.8.

This figure shows that for all 3 problems, AUC values improve steadily with increasing N, and almost reach a plateau at $N = 5$. It conforms with our finding in Section 4.2.3 that the sequentializaed models are dominated by the most recent events.

4.3 Conclusion

4.3.1 Comparison with event-based patient level decision trees

In this chapter, event-based patient level decision trees were built for comparison with our sequentialization approach. We saw improvements over conventional event-based approach for cost and LOS predictions but not 30-day readmission. It shows that for some problems that have strong sequential patterns, sequentialization approach can do better than event-

based approach. But for problems that have weak sequential patterns, pure sequentialization approach may not be helpful. It leads to our exploration of combined approach next chapter.

4.3.2 Representation power of the sequentialization approach

As seen from the sample sequentialized trees in Figure 4.5 and 4.6, the models can capture general patterns of different elements in a sequence with respect to the response variable, with no underlying assumptions on how the patterns may look like. Since we can apply any kind of supervised machine learning technique on the sequentialized tables, there is no limit on the type of sequence patterns that can be represented by this approach.

4.3.3 Limitations

In spite of having an efficient deployment phase, the training phase of this approach is highly computational intensive. It involves building large number of machine learning models and optimizing large number of parameters. The journey to find the best machine learning technique for each feature may involve many rounds of trial and error. It is also very space inefficient. It expands every feature of each sequence in the training data into every possible subsequences during the construction of sequentialized tables. To get around this issue we limited the maximum length of subsequences to 20 and number of features to 4, which is believed to have greatly limited the predictive power of this approach. Fortunately, most of the tasks in the training phase are independent of each other and are thus parallelizable. With the abundance of low-cost processing power and storage spaces in cloud services, the scalability issue is solvable.

Chapter 5

THE COMBINED APPROACH

5.1 Motivation

The sequentialization approach introduced in Chapter 4 harnesses the sequential (vertical) patterns of each feature in the data with respect to the response variable. It does not take into account the interaction of features within a single event (horizontal). The idea is illustrated by the red arrows in Table 5.1.

On the other hand, conventional event-based approach harnesses the interaction of different features in a single event (horizontal). It does not take into account the sequential (vertical) patterns of features within a sequence, which could be a rich source of information. The idea is illustrated by the blue arrows in Table 5.1.

We have seen in Chapter 4 that the sequentialization approach worked better for some problems (cost and length of stay prediction), while event-based approach worked better for other problem (30-day readmission prediction). What if we combine them? Would that allow us to obtain even better results than with each technique individually? That is the research question we address in this chapter.

5.2 Main Algorithm

The combined algorithm is divided into training, validation, and testing phases. Assume the data has gone through necessary preprocessing such as cleaning and feature engineering. Split the data into training, validation and testing sets.

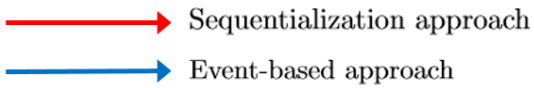
The training phase of the proposed algorithm is described as follows:

1. Train an event-based model for all features

   With the training data, train a conventional event-based model with any chosen supervised machine learning technique to predict y. The model is trained with all selected features.

Table 5.1: Sequentialization approach VS Event-based approach

Sequence Id	Feature x_1	...	Feature x_m	Response Variable y
s_1	a	...	β	y_1
s_1	a	...	γ	y_3
s_1	b	...	δ	y_2
s_1	b	...	α	?
s_2	b	...	β	y_1
s_2	a	...	δ	y_3
s_2	b	...	β	y_3
s_2	a	...	γ	?



2. Construct sequentialized tables for each feature

   For the training data, construct a new, separate table for each feature x by extracting all subsequences of the column for x in the original table, and associating them with the value of their response variable from the original table.

3. Divide each sequentialized table into subtables

   For each sequentialized table, divide it into subtables according to the length of sequence.

4. Train machine learning models for each subtable

   For each subtable, apply well-known supervised machine learning techniques to predict y.

The steps to make predictions on new data are described as follows:

1. Sequentialize new data for each feature

   When new data comes in as a sequence of the form of Table 4.1, extract the sequence for each of the m features.

2. Make a prediction for each sequentialized feature

   For each feature sequence, locate the model with the corresponding sequence length and apply the model to make a prediction.

3. Make a prediction with the event-based model on the new data. Let us denote the prediction by p_e.

4. Combine the predictions of each feature and event-based model in an ensemble

   After using all feature sequences and the event-based model to make predictions, combine them in an ensemble to obtain the final prediction. Let us denote the prediction and weight of the i-th feature by p_i and w_i, and the weight of the event-based model by w_e. The final prediction P is obtained by the formula,

$$P = p_1 w_1 + p_2 w_2 + \ldots + p_m w_m + p_e w_e \tag{5.1}$$

   where

$$\sum_{i=1}^{m} w_i + w_e = 1 \tag{5.2}$$

The weights are determined during the validation phase. The validation phase is described as follows:

1. Unweighted ensemble

   For an unweighted ensemble, simply assign equal values of $1/(m+1)$ to all weights.

2. Weighted ensemble

   For a weighted ensemble, the weights have to be optimized with regard to AUC, accuracy, or any other criteria. The weights can be optimized with any optimization algorithm by applying the above prediction steps to the validation data repeatedly.

After getting the weights, the ensemble can be tested by applying the above prediction steps to the testing data. Figure 5.1 shows the overall workflow of the combined approach.

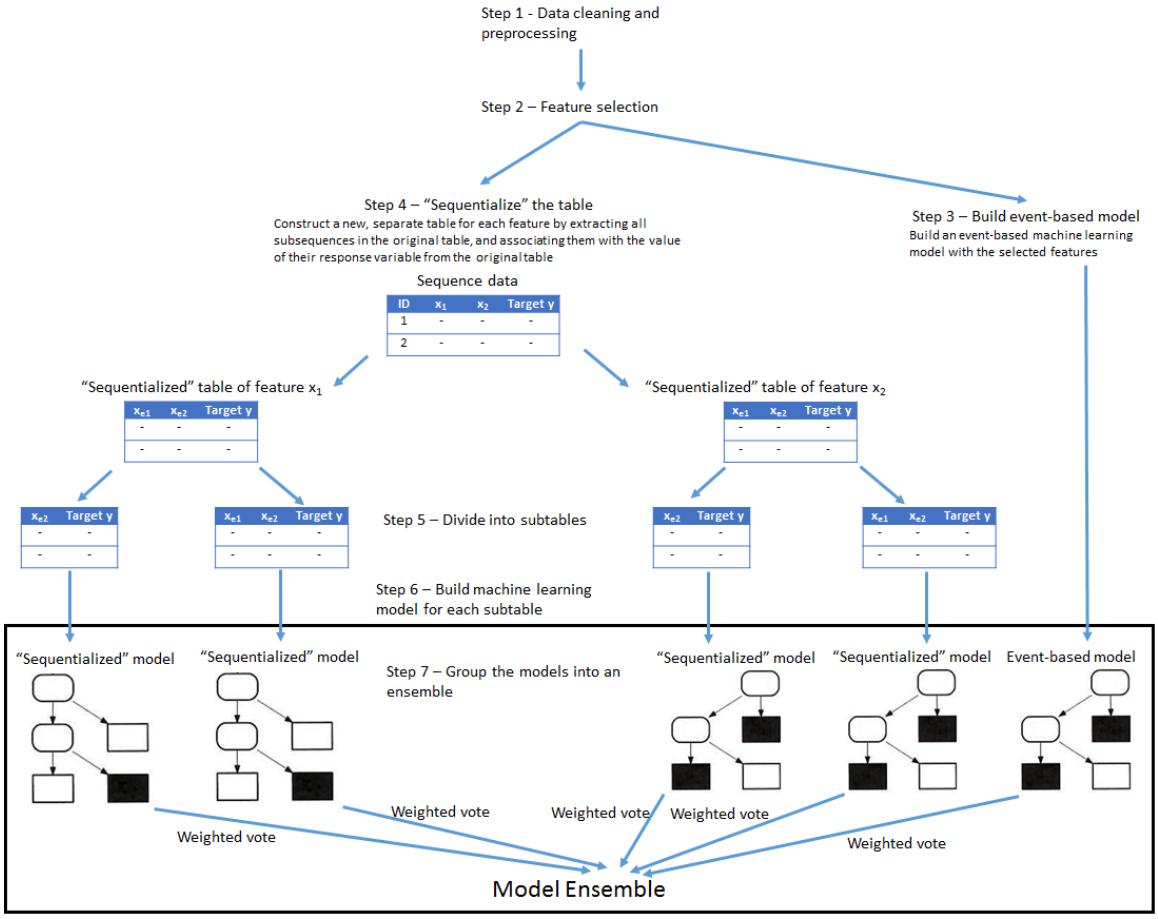


Figure 5.1: Overall workflow of the combined approach

5.3 Evaluation with OSHPD data

We evaluate the combined approach by applying it to the OSHPD data to predict 30-day readmission, cost and length of stay. The data has gone through all the cleaning and preprocessing steps as described in Chapter 2. The same feature selection and binarization of cost and length of stay is applied as in Chapter 3. The data is divided into 60% training, 20% validation and 20% testing.

5.3.1 Event-based decision tree

The same baseline decision trees that were built in Section 4.2.1 are reused as event-based models. These trees were built on the training data, for all 35 features. There are 3 such trees, i.e. one for each of the prediction tasks.

5.3.2 Sequentializating the table, training decision trees, and second feature selection

These steps are done exactly the same way as described Section 4.2.

5.3.3 Parameter tuning

The same R script in Section 4.2.6 is used for parameter tuning. However, now, for each problem, 5 weights are optimized with 4 weights for 4 selected features and 1 weight for the event-based model. The optimized weights for each problem are shown in Table 5.2, 5.3, and 5.4. The optimized weights are persisted into disk for further evaluation with testing data.

For 30-day readmission, Table 5.2 shows a drastic change of weights with the introduction of the event-based model. The event-based model has the highest weight of 0.4450. It can be explained from the results in Table 4.8 that 30-day readmission exhibits strong event-based pattern and weak sequential pattern.

We can also see substantial weight changes for cost and length of stay prediction in Table 5.3 and 5.4. The event-based models for these 2 problems do not have the highest weights. It can be explained from the results in Table 4.8 that cost and length of stay exhibits weak event-based pattern and strong sequential pattern.

Table 5.2: Optimized weights for sequentialized models based on 4 features and event-based model of 30-day readmission based on 35 features. Weights from Chapter 4 are replicated here for comparison.

	Length of Stay	Disposition	ED Visits in 6 Months	LACE Score	Event-based Model
Ch. 5	0.1355	0.2532	0	0.1661	0.4450
Ch. 4	0.2500	0.2500	0.2500	0.2497	-

Table 5.3: Optimized weights for sequentialized models based on 4 features and event-based model of Cost based on 35 features. Weights from Chapter 4 are replicated here for comparison.

	Cost	Length of Stay	Source of Admission	LACE Score	Event-based Model
Ch. 5	0.4139	0	0.2282	0.0756	0.2821
Ch. 4	0.2561	0.2392	0.2565	0.2480	-

Table 5.4: Optimized weights for sequentialized models based on 4 features and event-based model of LOS based on 35 features. Weights from Chapter 4 are replicated here for comparison.

	Cost	Length of Stay	Disposition	LACE Score	Event-based Model
Ch. 5	0	0.3525	0.2440	0.1661	0.2372
Ch. 4	0	0.3912	0.2180	0.3906	-

Table 5.5: AUC of unweighted combined approach with 35 features and 4 features, unweighted combined approach with 4 features, and weighted combined approach with 4 features. AUC values from Chapter 4 are replicated here for comparison.

	30-day	Cost	LOS
Event-based 35 features (Ch.4)	0.6156	0.5842	0.5808
Event-based 4 features (Ch.4)	0.6295	0.5835	0.5801
Unweighted Sequentialization 35 features (Ch.4)	0.5998	0.5705	0.5658
Unweighted Sequentialization 4 features (Ch.4)	0.6241	0.5897	0.5915
Weighted Sequentialization 4 features (Ch.4)	0.6241	0.5899	0.5979
Unweighted Combined 35 features (Ch.5)	0.6015	0.5784	0.5747
Unweighted Combined 4 features (Ch.5)	0.6292	0.5921	0.5939
Weighted Combined 4 features (Ch.5)	**0.6339**	**0.5945**	**0.5984**

5.3.4 Testing phase

3 evaluations were done with testing data for each problem. The first one is an unweighted combined approach with 35 features. There are 20 decision trees for each feature. Combined with the event-based decision tree, there are totally 701 trees in the ensemble. The second one is an unweighted combined approach with 4 features. The last one is a combined approach with optimized weights on 4 features. The latter two approaches have 20 decision trees for each feature. Combined with the event-based decision tree, there are totally 81 trees in the ensemble. The resulting AUC values are shown in Table 5.5.

5.3.5 Observations

Increases in AUC

As seen from Table 5.5, all AUC values from combined approaches showed improvement compared with their sequentialization counterpart. Therefore, adding an event-based model to the model ensemble actually improves predictive power, regardless of number of features

or whether it is weighted.

Importance of the second feature selection

The AUC values for all 3 problems increased significantly when the number of features is reduced from 35 to 4. This is because most of the features do not exhibit strong sequential patterns with respect to the 3 problems, as seen from Table 4.4. So including all features and having equal weights for all of them greatly reduced the contribution of the few most relevant features.

5.3.6 Importance of weights optimization

An increase in AUC values is observed for all 3 problems when weights are optimized. This is due to the fact that not all features are equally important when making predictions for the 3 problems. When combining with the event-based model, there is no way we can tell how much sequential patterns the problem exhibits before the optimization. For problems with little or no sequential patterns, the weight of the event-based model should be higher. For problems with stronger sequential patterns, the weights of the sequentialization models should be higher. Therefore, using validation data to optimize weights is the only way we can tune these weights to best fit the distribution and patterns of the data.

It is arguable whether the increase of AUC is worth the effort to optimize the weights, especially when the number of weights is high. As seen from Table 5.5, the increase of AUC for the combined approaches are 0.47%, 0.24%, and 0.45% for the 3 problems. And those for the sequentialization approaches are 0%, 0.2% and 0.64%. These are relatively small numbers. However, the degree of improvement varies for different types of data and problems. For example, if the sequential patterns of the problem vary greatly from feature to feature, or if the data has strong event-based patterns and weak sequential patterns, optimizing the weights will greatly increase the AUC. So it all depends on the type of data and problems we are working with. Nevertheless, optimizing weights almost guarantees an increase in AUC, as long as the testing data follows the same pattern and distribution of validation and training data. Therefore, if we are aiming at the highest possible AUC,

optimizing weights is an essential step.

5.4 Conclusion

In this chapter, we proposed a combination of an event-based and a sequentialization approach in a model ensemble. The proposed approach is tested with OSHPD data on 3 problems (30-day readmission, cost, and length of stay). The results are compared with the pure event-based approach and the pure sequentialization approach in Table 5.5. The combined approach outperformed the latter 2 in AUC values, and is the best results we have so far. Therefore, combining an event-based model with sequentialization models in an ensemble actually improved the predictive power of both in the case of OSHPD data. However, it is very important to use validation data to optimize the weights of the models. This is because different types of data have different patterns and distributions, and are not guaranteed to have the same kind of improvement with an average weight for all models.

Chapter 6

CONCLUSION

In this thesis, we presented a new sequentialization of features approach for making predictions about next events in complex event sequences. We evaluated our approach with hospital discharge data from the California Office of Statewide Health and Planning Development (OSHPD) on 3 problems, prediction of risk of 30-day readmission, cost and length of hospital stay. We compared the predictive performance with a conventional event-based approach, and found that the sequantialization of features approach has better results for cost and length of stay prediction, while the event-based approach has better results in 30-day readmission.

We also presented a combination of our sequentialization of features approach with the event-based approach. The combined approach is evaluated with OSHPD data on the same 3 problems. The results are compared with those of the sequentialization of features approach and the event-based approach being done individually. The combined approach shows the best performance among the 3 approaches in all 3 problems.

We discussed the limitations of the proposed approach. The training phase of this approach is both computationally intensive and space demanding. These difficulties are remedied by using parallel computing and cloud storage.

The work in this thesis opens many new directions for future research. Regarding the OSHPD data, additional experiments can be done by taking more features for sequentialization. The decision trees can also be replaced by any other classification or sequence prediction techniques that best fit the data.

The proposed approach for complex event sequence prediction is domain independent. It can be applied to any other applications where data can be represented as complex event sequences. One such example is retail, where the purchase history of a customer can be represented as a sequence of visits. We hope that by introducing the sequentialization of

features approach, the gap between complex event sequence prediction and conventional event-based and simple sequence prediction can be bridged, and more research interest in this area can be aroused.

BIBLIOGRAPHY

[1] Kirkwood F Adams, Gregg C Fonarow, Charles L Emerman, Thierry H LeJemtel, Maria Rosa Costanzo, William T Abraham, Robert L Berkowitz, Marie Galvao, Darlene P Horton, ADHERE Scientific Advisory Committee, Investigators, et al. Characteristics and outcomes of patients hospitalized for heart failure in the United States: rationale, design, and preliminary observations from the first 100,000 cases in the acute decompensated heart failure national registry (adhere). *American Heart Journal*, 149(2):209–216, 2005.

[2] Jayshree Agarwal. Predicting risk of re-hospitalization for congestive heart failure patients. Master's thesis, University of Washington, 2013.

[3] Leo Breiman, Jerome Friedman, Charles J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Wadsworth Publishing Company, 1984.

[4] Marshall H Chin and Lee Goldman. Correlates of early hospital readmission or death in patients with congestive heart failure. *The American Journal of Cardiology*, 79(12):1640–1644, 1997.

[5] Mukund Deshpande and George Karypis. Evaluation of techniques for classifying biological sequences. In *Advances in Knowledge Discovery and Data Mining*, pages 417–431. Springer, 2002.

[6] Omer Duskin and Dror G Feitelson. Distinguishing humans from robots in web search logs: preliminary results using query rates and intervals. In *Proceedings of the 2009 workshop on Web Search Click Data*, pages 15–19. ACM, 2009.

[7] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[8] Anika L. Hines, Marguerite L. Barrett, H. Joanna Jiang, and Claudia A. Steiner. Conditions with the largest number of adult hospital readmissions by payer. *HCUP Statistical Brief*, 172, 2011.

[9] Chun Pan Hon, Mayana Pereira, Shanu Sushmita, Ankur Teredesai, and Martine De Cock. Risk stratification for hospital readmission of heart failure patients: A machine learning approach. In *Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 491–492. ACM, 2016.

[10] 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.

[11] Harlan Krumholz, Sharon-Lise Normand, Patricia Keenan, Zhenqiu Lin, Elizabeth Drye, Kanchana Bhat, Yongfei Wang, Joseph Ross, Jeremiah Schuur, Brett Stauffer, et al. Hospital 30-day heart failure readmission measure methodology. *Report prepared for the Centers for Medicare & Medicaid Services*, 2008.

[12] Harlan M Krumholz, Eugene M Parent, Nora Tu, Viola Vaccarino, Yun Wang, Martha J Radford, and John Hennen. Readmission after hospitalization for congestive heart failure among medicare beneficiaries. *Archives of Internal Medicine*, 157(1):99–104, 1997.

[13] Terran Lane and Carla E Brodley. Temporal sequence learning and data reduction for anomaly detection. *ACM Transactions on Information and System Security (TISSEC)*, 2(3):295–331, 1999.

[14] Xuan Liu, Pengzhu Zhang, and Dajun Zeng. Sequence matching for suspicious activity detection in anti-money laundering. In *Intelligence and Security Informatics*, pages 50–61. Springer, 2008.

[15] Naren Meadem, Nele Verbiest, Kiyana Zolfaghar, Jayshree Agarwal, Si-Chi Chin, and Senjuti Basu Roy. Exploring preprocessing techniques for prediction of risk of readmis-

sion for congestive heart failure patients. In *Data Mining and Healthcare (DMH), at International Conference on Knowledge Discovery and Data Mining (KDD)*, volume 150, 2013.

[16] Andrew Y. Ng and Michael I. Jordan. 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*, pages 841–848. MIT Press, 2001.

[17] William W Parmley. Pathophysiology and current therapy of congestive heart failure. *Journal of the American College of Cardiology*, 13(4):771–785, 1989.

[18] Hude Quan, Vijaya Sundararajan, Patricia Halfon, Andrew Fong, Bernard Burnand, Jean-Christophe Luthi, L Duncan Saunders, Cynthia A Beck, Thomas E Feasby, and William A Ghali. Coding algorithms for defining comorbidities in icd-9-cm and icd-10 administrative data. *Medical Care*, pages 1130–1139, 2005.

[19] 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 *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[20] Pang-Ning Tan and Vipin Kumar. Discovery of web robot sessions based on their navigational patterns. In *Intelligent Technologies for Information Analysis*, pages 193–222. Springer, 2004.

[21] Wil MP Van der Aalst, Boudewijn F van Dongen, Joachim Herbst, Laura Maruster, Guido Schimm, and Anton JMM Weijters. Workflow mining: a survey of issues and approaches. *Data & Knowledge Engineering*, 47(2):237–267, 2003.

[22] Janice M Vinson, Michael W Rich, Jane C Sperry, Atul S Shah, and Timothy McNamara. Early readmission of elderly patients with congestive heart failure. *Journal of the American Geriatrics Society*, 38(12):1290–1295, 1990.

[23] Li Wei and Eamonn Keogh. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 748–753. ACM, 2006.

[24] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. A brief survey on sequence classification. *ACM SIGKDD Explorations Newsletter*, 12(1):40–48, 2010.

[25] Bichen Zheng, Jinghe Zhang, Sang Won Yoon, Sarah S. Lam, Mohammad Khasawneh, and Srikanth Poranki. Predictive modeling of hospital readmissions using metaheuristics and data mining. *Expert Systems with Applications*, 42(20):7110–7120, 2015.