INTELLIGENT PATIENT DATA GENERATOR

by
Mojtaba Zare
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Health Services Research

Committee:

Janusz Wojtusiak, Ph.D., Chair

Sanja Avramovic, PhD., Committee Member

Andrew Crooks, Ph.D., Committee Member

Pj Maddox

PJ Maddox, Ph.D., Department Chair

Germaine Louis

Germaine M. Louis, Ph.D., Dean and Professor, College of Health and Human Services

Date: 7/23/20

Summer Semester 2020
George Mason University
Fairfax, VA

Intelligent Patient Data Generator

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

by

Mojtaba Zare
Master of Science
Universiti Teknologi Malaysia, 2015
Bachelor of Science
Babol Noshirvani University of Technology, 2011

Director: Janusz Wojtusiak, Associate Professor of Health Informatics
Division Director, Programs in Health Informatics
Director, Machine Learning and Inference Laboratory
Department of Health Administration and Policy
College of Health and Human Services

Summer Semester 2020
George Mason University
Fairfax, VA

DEDICATION

To my parents who supported me throughout different stages of my study from an earlier age. And to my Ph.D. advisor, Dr. Janusz Wojtusiak, who shared his valuable knowledge with me and patiently guided me in every steps of the way.

ACKNOWLEDGEMENTS

I would like to deeply thank Dr. Janusz Wojtusiak, without whom I would not have been able to complete this dissertation. As an advisor and dissertation director, he guided me with possible solutions and recommendations throughout the difficult stages of this dissertation. In addition to sharing his knowledge and expertise, he also arranged financial support for me during my studies. I would also like to thank my dissertation committee, Dr. Andrew Crooks and Dr. Sanja Avramovic, who improved this work by their thoughtful comments. I would like to thank many people who directly or indirectly contributed to this dissertation. I also thank faculty of College of Health and Human Services who taught different courses I had the privilege of taking. Finally, I thank my friends who were supportive and encouraged me along the way.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS

Area Under the ROC	AUC
Back Window Size	BWS
Clinical Classifications Software	CCS
Current Procedural Terminology	CPT
Current Procedural Terminology	CPT
Electronic Health Record	EHR
Elixhauser	ELIX
Forward Window Size	FWS
Health Insurance Portability and Accountability Act	HIPAA
Healthcare Common Procedure Coding System	HCPCS
Healthcare Cost and Utilization Project	HCUP
Intelligent Patient Data Generator	IntPDG
International Classification of Diseases, Tenth Revision	ICD-10
Machine Learning	ML
National Cancer Institute	NCI
National Provider Identifier	NPI
Performing National Provider Identifier	P_NPI
Publicly Available Data Approach to the Realistic Synthetic EHR	PADARSER
Receiver Operating Characteristics curve	ROC
Referring National Provider Identifier	R_NPI
Surveillance, Epidemiology, and End Results	SEER
Unified Medical Language System	UMLS
Weighted Itemset Error	WIE
World Health Organization	WHO

ABSTRACT

INTELLIGENT PATIENT DATA GENERATOR

Mojtaba Zare, Ph.D.

George Mason University, 2020

Dissertation Director: Dr. Janusz Wojtusiak

Patient data are regarded as highly sensitive and protected by federal, state and local policies that make it available to only those who have been given access to protected health information. Synthetic data generation provides one possible solution to the issue of limited access, but at the same time, it is a key challenge in big data benchmarking that aims to generate application-specific datasets. In this dissertation, first, a comprehensive literature on synthetic data generation is presented which helps readers and practitioners in effectively adopting data generator approaches and provides an insight into its state-of-the-art. Next, a Machine Learning (ML)-based algorithm, Intelligent Patient Data Generator (IntPDG), is proposed to generate scalable patient claims data. In order to construct a model for generating high quality of patient data, two main elements including back window size and hyperparameters of different ML algorithms are investigated. Besides, a data evaluation measure, Weighted Itemset Error (WIE), is presented and used to evaluate the quality of the generated data in hyperparameter optimization. To generate claim level data from patient level data, patterns and data structures of actual patient claims data are

gathered and used in probabilistic models. Once the data generator method is constructed, it is tested on simulating Medicare carrier claims data, consisting of three datasets: patient demographic table, patient claim table, and patient line table. To add another layer of validation to the synthetic data, summary statistics of the generated datasets are compared with that of Medicare data and result confirms the consistency and validity of the simulated claims data. The developed data generator method can be used to generate any sizes and any types of claims data such as inpatient and outpatient claims data or can be extended to generate other medical data such as Electronic Health Records (EHR).

CHAPTER ONE: INTRODUCTION

In this chapter, an introduction to patient data, the issue of access to patient data, and realistic synthetic patient data as a solution to the issue are presented. Next, in Section 1.1, different types of patient data are discussed. Finally, different coding systems used in healthcare are explained in Section 1.2.

Obtaining datasets is costly in terms of the resources required and using those datasets usually leads to privacy problems, especially when it comes to the health domain and identifiable patient data (Esposito et al., 2018; Abouelmehdi et al., 2018; Kostkova et al., 2016). To address this issue, synthetic data can be generated and be substituted with real data in many applications.

Generating synthetic data is a key challenge in big data benchmarking and the problem of generating synthetic data is not new. Over the past years, a few studies conducted to generate EHR data (Walonoski et al., 2017; Buczak et al., 2010; Choi et al., 2017) and in one case generating patient image data (Guibas et al., 2017). To the best of author's knowledge, there is no study conducted yet to generate patient claims data specifically using ML-based algorithms proposed in this dissertation. Clinical data from EHR are critical for analyses to improve health care delivery. However, the use of claims data can effectively complement EHR data by providing an extremely broad view of a patient’s interactions across the continuum of the health care system, reduce selection bias and provide access to large and diverse samples (Stein et al., 2014).

In addition, generated synthetic data (such as EHR or Claims data) without complete documentation cannot be validated, which reduces the utility of many methods for the wider scientific community (Dube et al., 2013; Birkin et al., 2006; Stodden, 2010). In most of the previous studies, a clinician or only statistical information of data were mainly used to evaluate the generated data (Buczak et al., 2010; Choi et al., 2017; Walonoski et al., 2017); however, the validation needs to be done consistently with uniform measures applicable across datasets.

Hence, in this dissertation, a new algorithm, IntPDG (Intelligent Patient Data Generator) is proposed to generate scalable patient claims data. In order to evaluate quality of the generated synthetic data, an evaluation measure is presented to effectively evaluate the quality of the generated data. Therefore, the objectives of this dissertation are classified into two main categories:

I. Development and study of a novel ML-based algorithm for generating synthetic patient data.

II. Development and study of an evaluation measure for testing quality of synthetic data.

1.1. Types of Health Data

Healthcare data is a main resource for most health research and they are either collected during the course of ongoing patient care or as part of a formal clinical trial program. While health data have so many types, but, in general, health data can be categorized into five main groups as follows: Claims data, EHR, Patient/Disease registries, Health surveys, and Clinical trial data. There are several other types of health data that are out of scope of this dissertation.

1.1.1. Medical Claims

Claims data which is a type of administrative data describe the billable interactions (insurance claims) between insured patients and the healthcare delivery system. Claims data is a rich source of information that includes information related to diagnoses, procedures, and utilization. There are numerous analyses that can be conducted on claims data to derive information and knowledge to drive decision-making. Claims data can be used to compare services provided by specific providers or health care organizations based upon specific diagnoses (or combinations of diagnoses). It can also be used to evaluate quality of care provided by health care providers. For example, claims data can reveal whether a doctor followed nationally recommended medical protocols for treating patients diagnosed with diabetes. Claims data can also be used for population health analytics. For example, using the carrier claims data and determining high patient utilizers and so to examine their economic impact (Vestal, 2014).

Claims data is categorized into five general categories including: carrier, inpatient, outpatient, pharmacy (prescription drug events), and enrollment (Stein et al., 2014). The sources of claims data can be obtained from the government (e.g., Medicare) and/or commercial health firms (e.g., United HealthCare, Aetna). Claims data usually are not restricted to services delivered at only one particular medical center and they often capture a relatively diverse group of enrollees receiving care in various settings across larger geographic regions. Large sample sizes can be particularly useful for studying uncommon health conditions and findings can be generalizable (Obermeyer et al., 2017).

For most analyses using claim data, researchers can follow patients longitudinally to study use patterns, outcomes, costs of care and their changes over time (He et al., 2014; Gagne et al., 2011). Compared with most population-based cross-sectional studies (Varese et al., 2012; Hu et al., 2015) which capture the presence or absence of conditions at specific time points, claims data allow investigators to follow patients from their date of enrollment in a plan to their exit date or death. Furthermore, researchers can identify patients experiencing complications months or years after surgery without much loss to follow up because of receipt of care by a different provider, so long as enrollment in the plan is maintained. Researchers can also assess the temporal relationship among different conditions, procedures, or medications with respect to one another based on the date of service (Baowaly et al., 2018).

On the other hand, claims data have limitations. Claims data do not include information about patients' lab results, vital signs, patient surveys, habits (smoking, etc), and physician's notes. For example, if we are trying to identify smokers, we cannot depend on claims data to provide this information. If we want to track a patient's response to depression treatment, we need to see the results of a survey over time. Claims data does not afford us the opportunity to evaluate this (Wilson and Bock, 2012). There are also challenges with using claims data. One challenge related to using claims is assessing data quality and accounting for incomplete data. Other challenges include integrating data from multiple sources and developing methods for describing utilization of care or appropriateness of care (Stein et al, 2014). Technical challenges with creating specific datasets based upon claims data include: converting claims into unique visits, categorizing providers and locations of service, selecting the most useful measures of utilization and expenditures (Tyree et al., 2006).

1.1.2. Electronic Health Records

EHR is an electronic version of a patients' medical history maintained by a provider over time. EHR includes all of the key administrative clinical data relevant to a person's care under a particular provider, including demographics, progress notes, medications, vital signs, past medical history, immunizations, laboratory data and radiology reports (Jensen et al., 2012). EHR automates access to information and has the potential to streamline the clinician's workflow. EHR also has the ability to support other care-related activities directly or indirectly through various interfaces, including evidence-based decision support, quality management, and outcomes reporting. EHRs are the next step in the continued progress of healthcare that can strengthen the relationship between patients and clinicians. The data, and the timeliness and availability of it, will enable providers to make better decisions and provide better care (CMS, 2014; Ancker et al., 2014). For example, EHR can improve patient care by:

- Reducing the incidence of medical error by improving the accuracy and clarity of medical records.
- Making the health information available, reducing duplication of tests, reducing delays in treatment, and patients well informed to take better decisions.

- Reducing medical error by improving the accuracy and clarity of medical records.

EHR systems are designed to store data accurately and to capture the state of a patient across time. It eliminates the need to track down a patient's previous paper medical records and assists in ensuring data is accurate and legible. It can reduce risk of data replication, as there is only one modifiable file, which means the file is more likely up to date, and decreases risk of lost paperwork (Singh et al., 2015).

1.1.3. Patient / Disease Registries

A registry is a collection of information about individuals that tracks a narrow range of key data for certain diagnosis or condition such as Alzheimer's disease, cancer, diabetes, heart disease, or asthma. Many registries collect information about people who have a specific disease or condition, while other registries seek participants of varying health status who may be willing to participate in research about a particular disease. Individuals provide information about themselves to some registries on a voluntary basis, while reporting to others is mandatory by providers (i.e., commutable diseases) (NIH, 2020a).

Registries can be sponsored by a government agency, nonprofit organization, health care facility, or private company. Registries data allow health care professionals to improve treatment, and help researchers to design better studies on a particular condition, including development and testing of new treatments. Example of registries are Alzheimer’s Prevention Registry (Alzheimer’s Prevention Registry, 2020), Children's Health Foundation Pediatric Asthma Registry (Children’s Health Foundation, 2013), and the Surveillance, Epidemiology, and End Results (SEER) registries (SEER, 2020).

For example, SEER Program of the National Cancer Institute (NCI) began collecting data on cancer incidence and survival in the United States cases on January 1, 1973 and currently it covers cancer registers 34.6 percent of the U.S. population. The SEER program registries routinely collect data on patient demographics, primary tumor site, tumor morphology and stage at diagnosis, first course of treatment, and follow-up for vital status. The SEER Program is the only comprehensive source of population-based information in the United States that includes stage of cancer at the time of diagnosis and patient survival data (SEER, 2020).

1.1.4. Health Surveys

Health survey is defined as the ongoing systematic collection, analysis, and interpretation of health data, required for the design, implementation, and evaluation of health prevention programs of a population (Blackwellet al., 2001). Health survey generally include measures of risk factors, health behaviors, and non-health determinants or correlates of health such as socioeconomic status. The range of measures that can be included is wide and varies by survey. Age, gender, and race/ethnicity are the basic demographic variables that are included in health surveys (Schauer, 2015).

Example of health surveys are Behavioral Risk Factor Surveillance System health survey (CDC, 2019), Medicare Health Outcomes Survey (NIH, 2019) and Demographic and Health Surveys (DHS, 2020). For example, the Behavioral Risk Factor Surveillance System (BRFSS) is a state-based system of telephone health surveys that collects information on health risk behaviors, preventive health practices, and health care access primarily related to chronic disease and injury. The BRFSS survey was established in 1984.

Data in BRFSS are collected monthly in all 50 states, Puerto Rico, the U.S. Virgin islands, and Guam (CDC, 2019).

1.2. Coding System in Healthcare

Medical codes are used to describe doctor's diagnoses, prescriptions, procedures performed on a patient, determine costs, and reimbursements, which make up a crucial part of the medical claim. The Unified Medical Language System (UMLS), which is a compendium of many controlled vocabularies in the biomedical sciences, provides a mapping structure among these vocabularies and thus allows one to translate among the various terminology systems. The base of the UMLS consists of over 100 incorporated controlled vocabularies and classification systems resulting in over 1 million biomedical concepts and 5 million concept names. Some examples of the incorporated controlled vocabularies are CPT (Current Procedural Terminology) and ICD-10-CM (International Classification of Diseases, Tenth Revision, Clinical Modification) (NIH-UMLS, 2019). The different coding systems used in this dissertation are explained as follows.

1.2.1. ICD-10-CM

The ICD-10-CM is a system used by physicians and other healthcare providers to classify and code diagnoses and symptoms of patients in the United States. It provides a level of detail that is necessary for diagnostic specificity and morbidity classification in the U.S. ICD-10-CM is published by the World Health Organization (WHO) where unique alphanumeric codes are used to identify known diseases and other health problems. There are over 69,000 ICD-10-CM diagnosis codes. According to WHO, physicians, coders, health information managers, nurses and other healthcare professionals also use ICD-10-

CM to assist them in the storage and retrieval of diagnostic information. ICD records are also used in the compilation of national mortality and morbidity statistics. All Health Insurance Portability and Accountability Act (HIPAA)-covered entities must adhere to ICD-10-CM codes, as mandated by the U.S Department of Health and Human Services (HHS) since October 1, 2015 (CDC, 2015).

1.2.2. Healthcare Common Procedure Coding System

Healthcare Common Procedure Coding System (HCPCS) is based upon CPT (Current Procedural Terminology). In fact, the first level of HCPCS is identical to CPT. HCPCS was developed by the Centers for Medicare and Medicaid (CMS) for the same reasons that the American Medical Association developed Current Procedural Terminology codes (CPT): for reporting medical procedures and services. HCPCS codes are used to represent medical procedures to Medicare, Medicaid, and several other third-party payers. The code set is divided into three levels as follows (Pereira, 2020):

Level I: Level one HCPCS codes is identical to CPT, though technically those codes, when used to bill Medicare or Medicaid, are named as HCPCS codes. CMS looked at the established CPT codes and decided that they did not need to improve upon or vary those codes, so instead they folded all of CPT into HCPCS.

Level II: The second level of HCPCS codes are designed to represent non-physician services like ambulance rides, wheelchairs, walkers, other durable medical equipment, and other medical services that do not fit readily into Level I. Where CPT describes the procedure performed on the patient, it does not have many codes for the product used in the procedure. HCPCS Level II takes care of those products and pieces of medical equipment. In fact, the real difference between CPT and HCPCS comes in is in Level II of HCPCS and the HCPCS modifiers. Level II codes are, like Level I, five characters long, but Level II codes are alphanumeric, with a letter occupying the first character of the code. These codes, like those in ICD and CPT, are grouped together by the services they describe, and are in numeric order. One can generally refer to the range of codes by their initial character. J-codes, for example, are the codes for non-orally administered medication and chemotherapy drugs.

Level III: The third level of HCPCS codes are considered only as local codes and are not nationally accepted and only few insurances would accept reporting these codes. These codes represent an item or service, which is not, included in the HCPCS level I and level II codes. Normally these codes would starts with an alphabet X or Z followed by four numeric characters like HCPCS level II codes.

1.2.3. Elixhauser Index

The Elixhauser (ELIX) index is a method of grouping diagnosis codes of patients, which can be used as a useful way for categorizing diagnosis codes when exploring patient data and it also can serve as a tool for reporting statistical information. There are 29 ELIX categories; each includes a range of diagnosis codes. For example, Congestive Heart Failure in an ELIX index, which includes all diagnosis codes of patient, related to this group of disease such as coronary artery disease, high blood pressure, and disorders of the heart valves (Elixhauser et al., 1998). In this dissertation, ELIX index are used to group patients' diagnosis codes (ICD10-CM). Patients' diagnosis codes are grouped into 30 groups, where 29 are ELIX categories and “others” contains any ICD10-CM that is not in included in the 29 ELIX categories.

1.2.4. Clinical Classifications Software

The Clinical Classifications Software (CCS) was developed as part of the Healthcare Cost and Utilization Project (HCUP), a Federal-State-Industry partnership sponsored by the Agency for Healthcare Research and Quality (AHRQ). Two types of CCS codes exist, one type for categorizing diagnoses codes, and another type for categorizing procedures codes. In the presented work only CCS medical procedure codes were used which are 244 groups. CCS procedure code is a medical procedure categorization scheme that can be employed in many types of projects analyzing patient data on procedures. More than 10,000 CPT procedure codes are collapsed into 244 manageable number of clinically meaningful categories where in some situation they are more useful for presenting descriptive statistics than are individual procedure codes (HCUP, 2019).

CHAPTER TWO: LITERATURE REVIEW

A variety of synthetic data generation methods has been developed across different domains. In this chapter, the studies related to data generation method are grouped into two main categories and briefly explained. The two groups are: 1- data generator in biomedical field, and 2- data generator in non-biomedical field (see Figure 1).

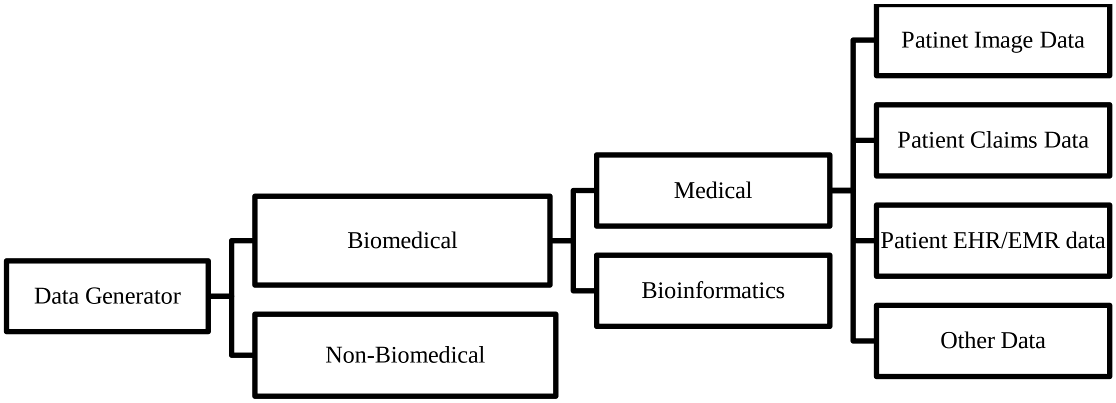

Figure 1 Classification of Data generator studies

2.1. Data Generation in Non-biomedical Fields

Several research studies and practical implementations have been done to generate synthetic data in domains other than biomedicine. For example, Kofinas et al. (2018) created a methodology to generate synthetic household water consumption data. More specifically, they designed an algorithm to generate flowrate records for households' water supply point. In order to do that, they captured statistical facts of the actual records and simulate those characteristics to generate the synthetic data. For example, the number of incidents per day, the duration of each incident, the time of the day most likely for an incident to occur and the flowrate of the event. In addition, summing all simulated water consumptions during a day and a year as whole should match the actual total water consumption. For validating the generated dataset, they produced curves of the probability of occurrence of an incident for both simulated and actual water consumptions; this way they validated the "incident occurrence" variable. Next, the generated flowrate values are validated by checking the fitting in simulating actual average flowrates. In addition, they compared the total generated water consumptions by the algorithm and actual water consumptions.

In another study conducted by Del Carmen et al. (2017), they claimed that the evaluation of Context-Aware Recommender Systems (CARS) is a challenge, due to the scarce availability of appropriate datasets, which incorporate contextual information related to the ratings provided by the users. Hence, they presented DataGenCARS, a Java-based synthetic data generator, which generated synthetic datasets of users, items, contexts, and ratings, that can be used to evaluate CARS. The key features of DataGenCARS is by designing different schemas such as user and context schemas which constrains domains, ranges, and types of each attributes. Next, they applied random probability distribution functions to generate attribute values. In order to evaluate the DataGenCARS, they performed two set of experiments. In the first experiment, they generated a synthetic dataset that tried to replicate an original dataset. After replicating the dataset, they compared the histograms, distributions and statistical properties of different attributes in the original and generated datasets. In the second experiment, they compared the performance of the recommendation algorithm with the original dataset (10 fold cross validation. In each fold, 70% of the data is used for training and 30% of the data is used for testing) vs the performance of the recommendation algorithm when it was trained using the synthetic dataset and tested using real dataset.

Kavak et al. (2019) stated that utilizing real-world Location-Based Social Network (LBSN) datasets in studies has major weaknesses including sparse and small datasets, privacy concerns, and a lack of authoritative ground-truth. Hence, they proposed a geo-simulation framework to simulate human behavior and to generate synthetic LBSN data that captures the location of users over time as well as social interactions of users in a social network. For proposing their data generation framework, Kavak et al. (2019) identified two main challenges: 1) To enrich the simulation with plausible human behavior by integrating psychological/social theories such as Maslow's hierarchy of needs (Maslow, 1943) and the theory of planned behavior (Ajzen, 1991). 2) The creation of a scalable and efficient geo-simulation design to accommodate millions of individuals to be simulated simultaneously. They claimed, a possible approach to implement the envisioned framework is to employ agent-based modeling; as an example the MASON (Multi-Agent Simulation of Neighborhoods) open-source simulation toolkit (Luke et al., 2005) and its GIS extension, GeoMASON.

2.2. Data Generation in Biomedical Field

Several methods have been proposed specifically for generating various types of biomedical data. As an example, Walonoski et al. (2017) presented Synthea, an open-source software intended for longitudinal simulation of synthetic patients. The software models the ten most frequent reasons for primary care encounters and the ten chronic conditions with the highest morbidity in the United States. Synthea uses the Publicly Available Data Approach to the Realistic Synthetic EHR (PADARSER) framework (Walonoski et al., 2017) in a top-down approach that generates synthetic EHR with coded entries in the Health Level-7 (HL7) Fast Healthcare Interoperability Resources (FHIR) standard format for the entire lifetime of the synthetic patient. Figure 2 presents the PADARSER framework.

The PADARSER Framework uses publicly available health statistics and assumes that access to the real EHR is impossible; it makes use of clinical guidelines or protocols in the form of care maps; and it employs methods to inherent realistic properties in the resulting synthetic EHR. In fact, clinical care maps and public health statistics are used to construct models of disease progression and treatment in a Generic Module Framework that encodes these models as state transition machines in an open and documented JSON format. The synthetic data validation was performed only on the data generated for type 2 diabetes (T2B). The methodology used for validation of the synthetic data was by comparing the properties (such as variance, distribution) of the generated data with publicly available data (Walonoski et al., 2017).

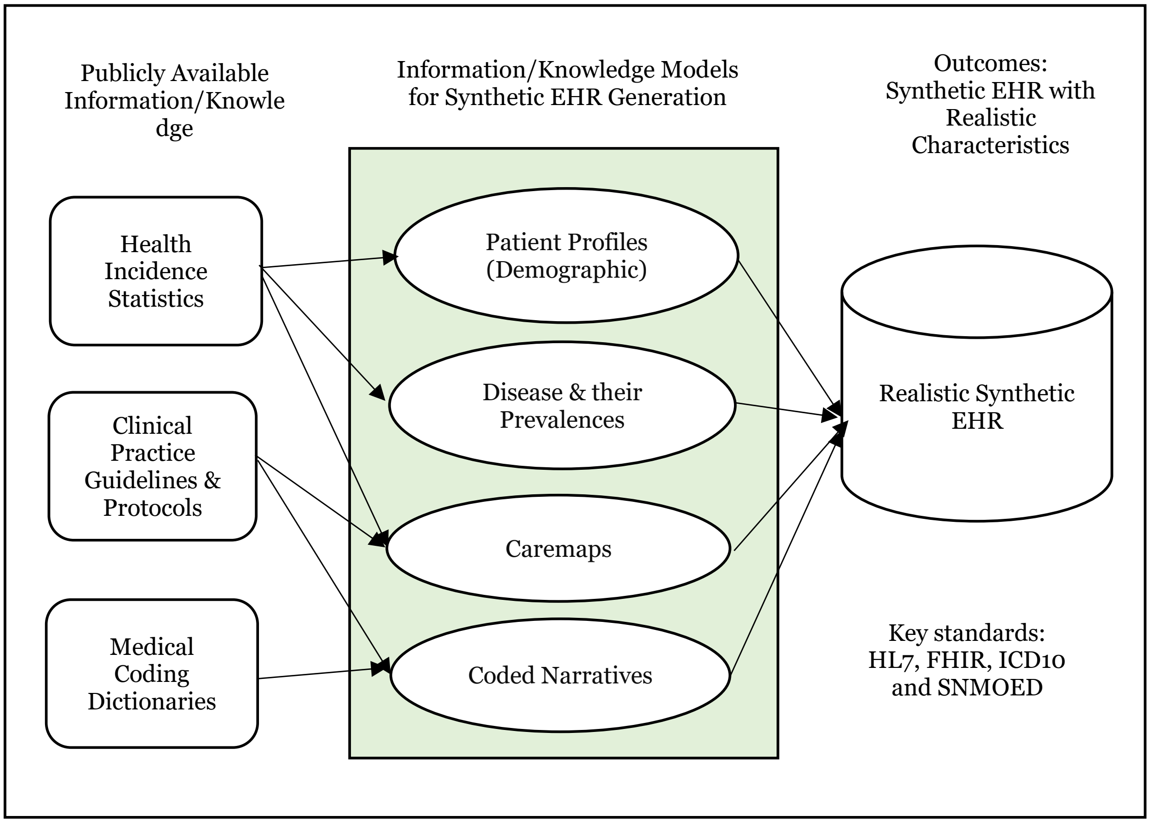
Publicly Available Information/Knowle dge
Information/Knowledge Models for Synthetic EHR Generation
Outcomes: Synthetic EHR with Realistic Characteristics
Health Incidence Statistics
Clinical Practice Guidelines & Protocols
Medical Coding Dictionaries
Patient Profiles (Demographic)
Disease & their Prevalences
Caremaps
Coded Narratives
Realistic Synthetic EHR
Key standards: HL7, FHIR, ICD10 and SNMOED

Figure 2 PADARSER as the conceptual framework for Synthea (based on Walonoski et al., 2017)

Buczak et al. (2010) developed a data-driven methodology called Synthetic Electronic Medical Records Generator (EMERGE). The method focuses on generating synthetic EMR data for disease outbreaks. Their method works in three main steps: synthetic patient demographics generation; identification of care patterns that for similar health problems; and application of the discovered care patterns to the synthetic patient population. In fact, this care pattern is defined as the sequence of health-care events that real patient experiences and it is used to create entries in the synthetic EMR. After a care pattern is identified from the care patterns present in the real EMR data set, the synthetic

EMR created based on/using statistical distribution. The generated data include visit records, clinical activity, laboratory orders/results and radiology orders/results. The authors applied the method to generate 203 synthetic tularemia outbreak patients. The validation of the generated synthetic data was done by a medical expert.

In another study conducted by Choi et al. (2017), the authors proposed a data generator approach called medical Generative Adversarial Network (medGAN) designed to generate realistic synthetic patient records. The method uses real patient records to generate high-dimensional discrete variables, including binary indicators and count features. It applies a combination of an auto-encoder and generative adversarial networks that are used to generate synthetic data. Quality of the synthetic was evaluated through qualitative and quantitative methodology. The qualitative analysis was performed through the help of a medical expert. The quantitative evaluation was performed using three datasets (Sutter Palo Alto Medical Foundation, MIMIC-III, and heart failure study datasets from Sutter) where distribution statistics and predictive modeling are used to compare the generated data with the real datasets.

Guibas et al. (2017) argued that medical imaging data is scarce, expensive, and fraught with legal concerns regarding patient privacy. In their study, they proposed a two-stage pipeline below for generating synthetic medical images from a pair of Generative Adversarial Networks (GAN). Stage-I GAN: This stage produces segmentation masks that represent the variable geometries of the dataset. The purpose of Stage-I is to generate varied segmentation masks. This stage is based on the deep convolutional generative adversarial network (DCGAN) architecture, and built on the TensorFlow platform. Stage-II GAN: This stage translates the masks produced in Stage-I to photorealistic images. The purpose of Stage-II GAN is to translate segmentation masks to corresponding photorealistic images. Stage-II GAN is also built on the TensorFlow platform. The model is based on an image-to-image translation network. In order to evaluate the reliability of the synthetic data, they used the synthetic data to train a u-net segmentation network. Next, they evaluated the trained u-net on test images from the DRIVE database and compared them with the ground truth by calculating an accuracy score (F1 score). They also calculated the variance between the synthetic and real datasets through a divergence score (Kullback–Leibler (KL) divergence score).

Maciejewski et al. (2009) conducted a study to generate synthetic Syndromic-Surveillance data for evaluating Visual-Analytics (VA) techniques. In fact, they developed a system that lets users generate non-aggregated synthetic data records from emergency departments (EDs), using derived signal components from the Indiana Public Health Emergency Surveillance System (PHESS). Generated data includes synthetic patient location and demographic information (age and gender), along with the ED chief complaint and chief-complaint classification. Their data generator methods include using seasonal decomposition of time series by loess (locally weighted regression), an analysis of population distribution using multiple kernel density estimation models, and an analysis of the age and gender of the populations and their correlation to chief complaints. The goal for creating time series data was to generate a time series of the number of patients a given ED sees daily.

In another study conducted by Baowaly et al. (2018), in order to generate EHR data, they modified Generative Adversarial Network (medGAN) (a type of neural networks) to obtain two synthetic data generation models - appointed as 1) medical Wasserstein GAN with gradient penalty (medWGAN) and 2) medical boundary-seeking GAN (medBGAN). They used two databases, MIMIC-III and National Health Insurance Research Database (NHIRD), Taiwan. First, they trained the models and generated synthetic EHRs by using these three models. Then, they analyzed and compared the models' performance by using statistical methods (Kolmogorov–Smirnov test, dimension-wise probability for binary data, and dimension-wise average count for count data) and 2 ML tasks (association rule mining and prediction). Result showed that their proposed models outperformed medGAN in all cases, and among the 3 models, boundary-seeking GAN (medBGAN) performed the best.

Wojtusiak (2016) presented a project discussing about large-scale generation of realistic synthetic patient data. He discussed that ML algorithms can be used to learn models from real data, combine these models with expert knowledge, and together apply to generate new synthetic data. In their project, they generated patient demographic information by creating probabilistic models and using statistics from US Census, US Center for Health Statistics, Social Security Administration and other sources.

Medicare Claims Synthetic Public Use Files (SynPUFs) were created to allow data analysts and software developers to become familiar with Medicare claims data. The data structure of the Medicare SynPUFs is similar to the CMS Limited Data Sets, while having a smaller number of variables and limiting to years between 2008 and 2010. The files have been created so that programs and procedures created on the SynPUFs will function on CMS Limited Data Sets as well. The purposes of the DE-SynPUF are to: 1) Allow data entrepreneurs to develop and create software and applications that may eventually be applied to actual CMS claims data; 2) Train researchers on the use and complexity of conducting analyses with CMS claims data prior to initiating the process to obtain access to actual CMS data; and 3) Support safe data mining innovations that may reveal unanticipated knowledge gains while preserving beneficiary privacy (CMS, 2020).

Dash et al. (2019) developed a method for generated human sleep patterns using a publicly available health dataset. The dataset used in their study is American Time Use Survey (ATUS) conducted by the U.S. Census Bureau. They first gathered summary statistics of the real data to characterize the events for a fixed set of time intervals. Next, a generative adversarial network is trained to generate synthetic data for human sleep patterns. Finally, they evaluate the generated data empirically without applying a consistent data evaluation method.

Goncalves et al. (2020) used and compared the data generator approach in the literature to generate EHR data for breast, lymphoma and leukemia, and respiratory cancer using real patient data. A subset of SEER data were used for training models and experimental analysis which consists of breast, lymphoma and leukemia, and respiratory cases diagnosed between 2010 and 2015, including around 360,000 patients. The process of data generation for each method was learning a model using SEER data first, and then generating synthetic EHR samples using the learned model. Independent Marginals (which is based on sampling from the empirical marginal distributions of each variable), Bayesian

Network, nonparametric Bayes approach, categorical latent Gaussian process (which is a generative model for multivariate categorical data) (Gal et al., 2015), and generative adversarial networks based model (Camino et al., 2018) were used as the main methods for generating synthetic EHR cancer data. To measure the quality of the synthetic data, two set of metrics were used: 1) Data utility: Statistical properties of the real data compared with synthetic data; 2) Information disclosure: Since real patient data were used to generate synthetic data, they used this metric to measure how much of the real data may be revealed by the synthetic data. Based on their defined evaluation metrics, Bayesian Networks, nonparametric Bayes approach and latent Gaussian approach have shown to be capable for EHR data generation because 1) statistical properties of the generated data were in line with real data, and 2) private information leakage from the model was not significant. The generative adversarial network-based model were not capable of generating realistic EHR samples. A summary of data generator methods and different types of data generated by previous studies are presented in Table 1.

Table 1 Previous studies related to synthetic data generation

Data generator method	Non-Biomedical	Image Data	Claim Data	EHR Data	Other patient data
Capturing statistical facts of the actual records and simulating those characteristics to generate water consumption data	Kofinas et al. (2018)				

Designing different schemas such as user and context schemas which constrains domains, ranges, and types of each attributes for generating contextual rating data provided by users	Del Carmen et al. (2017)				
Proposing a geo-simulation framework to simulate human behavior and to generate synthetic LBSN data that captures the location of users over time as well as social interactions of users in a social network. They claimed, a possible approach to implement the envisioned framework is to employ agent-based modeling; as an example the MASON (Multi-Agent Simulation of Neighborhoods) open-source simulation toolkit and its GIS extension, GeoMASON	Kavak et al. (2019)				
Clinical care maps and public health statistics are used to construct models of disease progression and treatment in a Generic Module Framework				Walonoski et al. (2017)	
Identification of care pattern in the real data and using statistical distribution to generate synthetic data				Buczak et al. (2010)	
A combination of an auto-encoder and generative adversarial networks				Choi et al. (2017)	

Deep convolutional generative adversarial network built on the TensorFlow platform		Guibas et al. (2017)			
Modified Generative Adversarial Network				Baowaly et al. (2018)	
Probabilistic models and using statistics from US Census, US Center for Health Statistics, Social Security Administration and other sources to generate patient demographic information				Wojtusiak , (2016)	
locally weighted regression & analysis of population distribution using multiple kernel density estimation models				Maciejewski et al. (2009)	
Not mentioned			CMS, (2020)		
Statistics of the real data are gathered to characterize the events for a fixed set of time intervals. Next, a generative adversarial network is trained to generate synthetic data for human sleep patterns					Dash et al. (2019)
Independent Marginals, Bayesian Network, nonparametric Bayes approach, categorical latent Gaussian process, and generative adversarial networks based model were for generating synthetic EHR cancer data				Goncalves et al. (2020)	

CHAPTER THREE: BACKGROUND METHODS

This chapter, which consists of five main sections, describes the main steps in model construction for generating synthetic patient data. In details, Section 3.1 describes the main idea of data generation method proposed in the dissertation. Medicate data using which the method is tested is explained in Section 3.2. Two main elements including back window size and hyperparameter optimization, which play key roles in the quality of constructed model, are discussed in Section 3.3 and 3.5 respectively. Section 3.4 explains the proposed data validation measure used to evaluate the quality of generated data in hyperparameter optimization.

3.1. General Framework: Model Construction and Data Generation

In this dissertation, the main concept in model construction for generating synthetic patient data is to use ML models to guide the generation process. ML-based models are constructed using real patient data for predicting “future” of a cohort of patients in the longitudinal data, and then using these models new data are generated iteratively. This can be described by a two-step process:

Step 1: Learn models for predicting patient’s “future”: Supervised learning methods can be applied to construct models for predicting events in the follow-up period for a given patient. One approach to do so is by using a sliding window approach in which the prediction horizon is shifted over time for the known patient data. Standard machine learning approaches can be constructed and iteratively applied to create data in following periods, D_2, D_3, … D_T. Specifically, $D_{T+1} = M(D_T)$, where $M = \{M_1, \ldots M_m\}$ represents a set of constructed ML models as shown in Figure 3.

Step 2: Generate synthetic data: Data generation needs to start with initialization of the first time period that is required to provide input attributes into prediction models created in Step 1, and initialize the data generation process. In the simplest approach, the initial data are real data. In a more advanced approach, taken in this study, the initial data are generated according to a certain distribution that includes desired population characteristics. After the initial data are created (D_1) in Step 2, models created in Step 1 are iteratively applied to create data in following periods (See Figure 3).

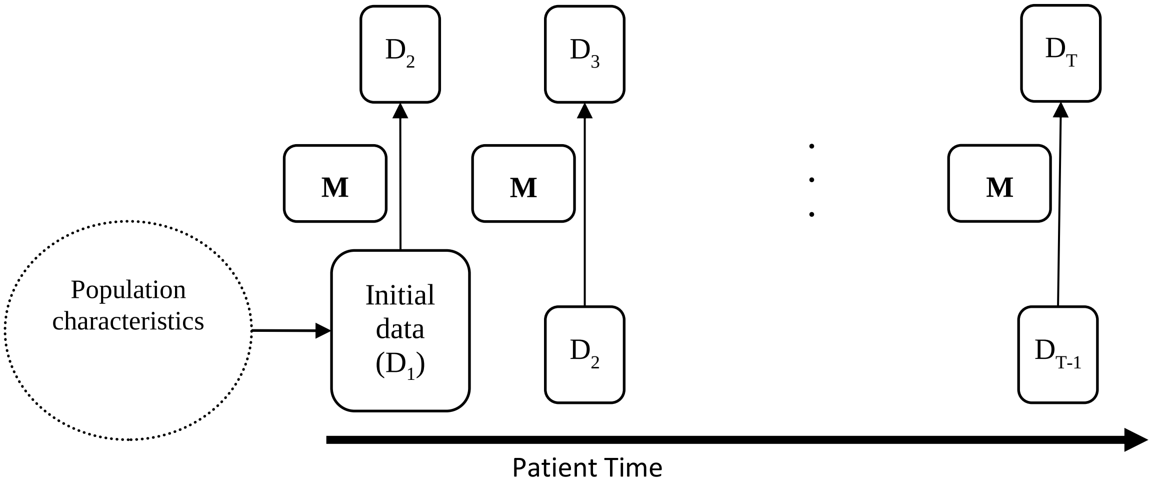

Figure 3 Main concept of generating synthetic data using ML models

One can immediately notice that synthetic data are used to generate more synthetic data. Intuitively, such process may diverge from real dataset further and further as the data are generated that may indicate bias in *M* that need to be tested.

3.1.1. How to Build a Good Model?

In the context of the presented work, good model is a model that can generate synthetic data with the highest quality. One can intuitively notice that models with high quality in terms of their accuracy also should generate high quality synthetic data. The quality of models depend on both data used for training those models as well as hyperparameters of used ML algorithms. Before applying the described model for patient data generation, two main elements need be investigated in the model construction.

i. The back window size of ML models needs to be investigated, since different back window size can contribute to different quality of the constructed model (discussed in Section 3.3).

ii. ML models with different set of hyperparameters need to be evaluated. Due to this fact, first an evaluation method for evaluating quality of the generated data needs to be presented (discussed in Section 3.4). Next, data are generated using different models and hyperparameters where the evaluation method is applied to choose the most tuned ML model (discussed in Section 3.5).

3.2. Simulate Medicare Patient Claims Data

Medicare is a federal health insurance program in the United States, begun in 1966. It primarily provides health insurance for Americans aged 65 and older, but also for some younger people with certain disabilities. Medicare has four part, Part A, B, C and D.

Medicare Part A covers cost of hospital inpatient care. Medicare Part B, on the other hand, covers outpatient medical care, preventive services, ambulance services, and durable medical equipment. The two programs, Part A and Part B, function as two halves of a comprehensive healthcare solution (CMS, 2019; NIH, 2020b).

Medicare Part C also known as Medicare Advantage Plan (or Medicare private health plan) is a type of health insurance plan that provides Medicare benefits through a private-sector health insurer. While the majority of people with Medicare get their health coverage from Original Medicare, some choose to get their benefits from a Medicare Advantage Plan. Medicare Advantage Plan contracts with the federal government and are paid a fixed amount per person to provide Medicare benefits. Medicare Part D, also called the Medicare prescription drug benefit, is an optional program to help Medicare beneficiaries pay for self-administered prescription drugs through prescription drug insurance premiums (the cost of almost all professionally administered prescriptions is covered under optional Part B). According to the 2019 Medicare Trustees Report, the Medicare Program is the second-largest social insurance program in the U.S., with 61.2 million beneficiaries and total expenditures of $796 billion in 2019. There are three main group of files in Medicare data: 1) Medicare Fee-For-Services claims, 2) Medicare Enrollment Beneficiary, which has demographic information of the beneficiaries, and 3) Medicare part D, which is prescription drug coverage and an optional benefit, offered to everyone who has Medicare. Medicare Fee-For-Services claims contain several tables including (Trustees Report & Trust Funds, 2019; CMS, 2019; NIH, 2020b):

- *Carrier Claims* (old file name Physician): Carrier claims contains claims submitted by non-institutional providers such as physicians, nurse practitioners, clinical laboratories, ambulance services, suppliers and stand-alone ambulatory surgical centers. Institutional files cover everything else including outpatient, inpatient, skilled nursing facility (SNF), hospice and home health agency.
- *Outpatient Claims*: Outpatient claims refers to claims for outpatient services performed by institutional providers in an outpatient setting. Examples of institutional outpatient providers include hospital outpatient departments, rural health clinics, renal dialysis facilities, outpatient rehabilitation facilities, comprehensive outpatient rehabilitation facilities, and community mental health centers.
- *Inpatient Claims*: Inpatient claims refer to claims performed by institutional providers in hospitals.
- *Medicare Provider Analysis and Review (MEDPAR)*: The MEDPAR file includes all Part A short stay, long stay, and skilled nursing facility (SNF) bills for each calendar year. MEDPAR contains one summarized record per admission.
- *Home Health Agency (HHA)*: The Home Health Agency file contains all claims for home health services.
- *Hospice*: The Hospice file contains claims data submitted by Hospice providers.
- *Durable Medical Equipment (DME)*: The Durable Medical Equipment contains final action claims data submitted to Durable Medical Equipment Regional Carriers (DMERCs).

In this dissertation, data will be generated for demographic table and carrier claims tables. Carrier claims consists of two tables: 1) carrier claim table, and 2) carrier line table. In carrier claim table, each row of the table belongs to a claim therefore claim number is unique in this table. Line table, on the other hand, is more detailed compared with carrier claim table where each claim in the line table can have multiple lines and each line belongs to a procedure performed on a patient.

Table 2, Table 3 and Table 4 show attributes for simplified versions of patient demographic table, carrier claim table and carrier line table respectively. Billing and payment information or any variables solely used for billing and payment purposes are excluded from these tables.

Table 2 Schema of demographic table of patients

Patient ID	Date of birth (dob)	Race	Sex	State code	Death date
1000001	10/10/1970	White	Female	10	N/A

Table 3 Schema of carrier claim table

Patient ID	Patient demographic (dob, race, sex, death date, state code)	Claim ID	Claim through date	National Provider Identifier (NPI)	Diagnosis codes (ICD 10) – 12 attributes
1000001	10/10/1970, Black, Female, N/A, 10	111	10/5/2010	1000000012	icd10_a, icd10b, icd10c

Table 4 Schema of carrier line table

Patient ID	Patient demographic (dob, race, sex, state code)	Claim ID	Claim line number	Claim through date	National Provider Identifier (NPI)	Claim Performing Physician Specialty Code	Primary diagnosis codes	Procedure code (CPT)
1000001	10/10/1970, Black, Female, N/A, 10	111	01	10/5/2010	1000000012	spec_a	icd10_a	hcpcs_a

Dataset: The dataset used in this dissertation is Limited Data Set Files (LDS) from 2012 to 2017 for about 44 million patients. Limited Data Set Files (LDS) are identical to the Medicare Beneficiary Encrypted Files, but they have been stripped of data elements that might permit identification of beneficiaries. These files contain beneficiary level health information but exclude specified direct identifiers as outlined in the Health Insurance Portability and Accountability Act (HIPAA Privacy Rule) (CMS, 2019).

Tools: In the presented work, PostgreSQL is used for data preprocessing. Python version 3.7 is used for all the data and experimental analysis, constructing prediction models, and data visualization.

3.3. Optimal Back Window Size of ML Models

Back window size of an ML model is the period preceding prediction time that is used to construct input attributes (Figure 19 in the next chapter shows how back window size is used to observe preceding prediction time and so predicting future). Back window size and training set size are two important elements in constructing high quality model for generating patient data. Two sets of experiments are conducted to find out the optimal back window size and the optimal size of the training set. It should be noted that all ICD10 diagnosis codes and HCPCS procedure codes are converted to ELIX codes (30 codes including “others”) and CCS (244 codes) respectively to decrease the complexity of analysis and improve efficiency of ML prediction models.

Experiment One: Model performance when evaluated on training sets. ML models are trained with different training set sizes and different back window sizes to evaluate the AUC of each trained model using each set of training set size and back window size. The prediction outcome of each model is one of the diagnoses or one of the procedures codes of the same training set with which the model is trained. The outcome of each model is observed for one month which is the month after back window horizon. The size of patients’ cohort in training sets changes from 1k, 5k, 10k, 15k, 20k, and 30k patients and back window size changes incrementally between 1 month and 22 months (each time 1 month is added to train a new model). Each time, size of back window is increased by 1 month until it becomes 22 months (660 days). Figure 4 shows the result of the first experiment where AUC is the average AUC of all the predicted outcomes (all ELIX and all CCS codes).

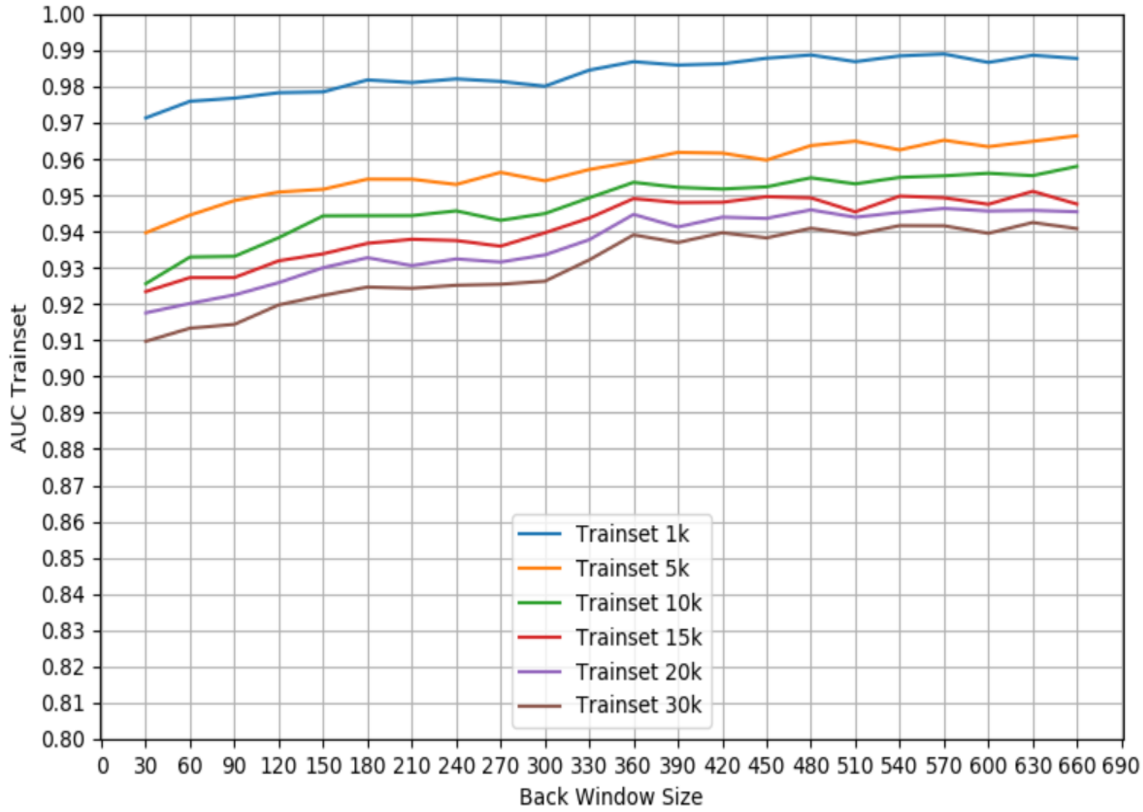

Figure 4 Model performance when evaluated on training sets

Experiment Two: Model performance when evaluated on testsets. The exact same ML models trained in the first experiment are used, but in the second experiment, an independent test set is created using which the performance of models are evaluated. The testset is 24 months longitudinal data for 5k patients. The testset is exclusive from any of the training sets meaning that there is no overlap between 5k patients in testset and patients in training sets. It is important to note that in the second experiment, the test set structure (the back window size and forward window size) should match with the structure of the training sets using which ML models are constructed so that the experiment would be valid. This means, for example, if a training set with 2 months back window size and 1 month forward window is created for training models, similarly a testset with 2 months back window size and 1 month forward window size is created using data for that 5k patients. Figure 5 shows the results of the second experiment where AUC is the average AUC of all the predicted outcomes (all ELIX and CCS attributes).

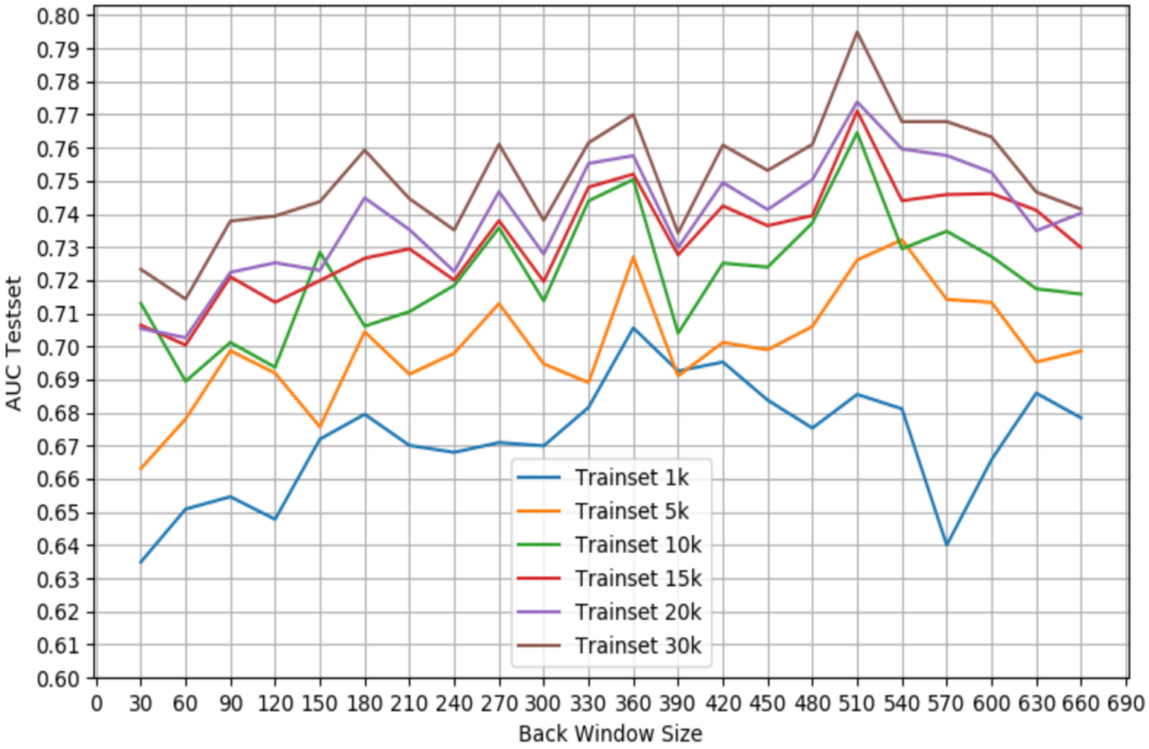

Figure 5 Model performance when evaluated on testsets

From both Figure 4 and Figure 5, it can be seen that models are not performing well for training set size below 10k (1k and 5k) patients. This is due to overfitting which means that the trained ML models predict well for the training sets (AUC above 0.9 for almost every back window size) but predict poorly for the testsets (AUC below 0.7 almost for every back window size). Training set size of 10k and above shows a considerable improvement in the AUC of models when evaluated on both training sets and testsets.

As can be seen from Figure 5, the highest value of the AUC when evaluated on testsets belongs to the training set size of 30k patients and the 17 months back window size. Therefore, the training set size of 30k patients with 17 months back window size are selected to train ML models for generating diagnosis and procedure attributes for patient level data in Section 4.2. It is important to note that AUC of the models is a key performance metric when prediction models are used to generate synthetic data. The actual predicted yes/no decision of the ML models is not important in generating data, but rather the probability that comes out of the models. Therefore, in these two experiments, AUC is considered as the key metrics over accuracy, precision or recall.

3.4. Synthetic Data Evaluation

Evaluation measures need to be applied to generated data in order to understand how good the quality of generated data is. Previous research normally used a medical expert or compared statistical information of generated data vs real data to evaluate the quality of generated data. However, it is a very time consuming task to evaluate big databases by a medical expert and using only statistical information (such as mean, mode, and standard deviation) for comparing datasets is not sufficient since the structures/properties within a dataset cannot be fully captured. Hence, there is a lack of evaluation measure that can be applicable across different datasets and objectively measure how realistic are generated synthetic data by capturing the structural information within a dataset.

How to evaluate the generated data objectively? One way to evaluate the generated data is to analyze properties of the generated data and compare these properties to those of a real dataset. It is important that the measured properties can be numerically quantified so that quality of datasets can be objectively compared. It is worth mentioning that measures such as accuracy (and it's variants precision, recall, AUC, etc.) are not applicable to the problem of evaluating synthetic datasets, because the generated data are different from the real data yet have to preserve certain properties, such as interaction between attributes and common itemsets found in both real data and generated data. For simplicity, let us assume in this section that the synthetic data consist of binary attributes. Such binary data are illustrated in Table 5.

Table 5 Example of binary data

ID	A_1	A_2	A_3	…	A_m
1	0	0	1		0
2	1	1	0		1
…				…	
N	0	0	0		1

The simplest test one can perform when comparing such synthetic data to the real data is to check frequencies of 1s and 0s for a single attribute as compared to the real dataset (and repeat the process for all attributes). While it is a good approach, it does not test for interactions between attributes and thus it omits important data characteristics. For example, if the data represent presence of diagnoses in patients within a given period, coded with a selected coding system, it is clear that the attributes are not independent from each other (i.e., one would expect heart failure more often among patients with diabetes). This approach can be easily generalized by considering pairs of attributes, thus counting frequencies of values co-occurring. Then, it may be even more accurate to find triples of attributes (rather than pairs) that are interrelated (i.e., in a medical diagnoses data it may be presence of diabetes, hypertension, and heart failure). Following this concept, one can compare frequency of arbitrarily large itemsets (sets of values present together) (Zare and Wojtusiak, 2018).

Formally, an itemset is defined as a set $I = \{i_1, i_2, \ldots i_s\}$ such that i_j is a value of attribute A_j (item). In other words, an itemset is a set of values of attributes, each value belonging to a specific attribute. Specifically, in the case of binary data, i_j indicates value 1 for A_j. Itemsets have a number of associated properties, with the most important being support, *sup*(*I*), defined as the probability of the itemset *I* being present in one row of data as shown in Equation 1.

Equation 1 Support of itemset I

$$sup(I) = \frac{\#rows\ containing\ I}{\#rows\ in\ data}$$

In order to evaluate the quality of the synthetic data (measuring the degree of similarity between the synthetic dataset and the real dataset) one needs to compare support of the itemsets in the real and synthetic data. It is reasonable to assume that larger itemsets (containing more items) are less important than smaller itemsets in comparing the data (i.e., in medical diagnosis data the discrepancy in frequencies for itemset {diabetes, heart failure, hypertension, renal disease} is less important than one for {diabetes, heart failure}) when calculating distance between datasets (error measure). Weighted Itemsets Error (WIE) measure presented in Equation 2 is developed in the presented work and it is an attempt to calculate such a distance between two datasets**.** To explain WIE formula, support of each itemset in the real data is subtracted from its counterpart in the generated data, then divides by m^{α} before it squares. In Equation 2:

- m is the size of that itemset.
- S_{IR_n} refers to value of support for Itemset index n in the Real data.
- S_{IS_n} refers to value of support for Itemset index n in the Synthetic data.
- Parameter α is the weight of the WIE measure that calibrates the method to measure the error.

Equation 2 Weighted Itemsets Error measure

$$\sum_{m=1}^{m} \frac{1}{m^{\alpha}} (S_{IR_n} - S_{IS_n})^2$$

The smaller the WIE, the more similar is the synthetic data to the real data, which is optimal. In addition, the bigger the WIE, the less similar is the generated data with the real data. The WIE measure is used as the main criterion for evaluating quality of binary data generated as part of the presented work and one step in generating synthetic claims data. Before the WIE measure is used for this purpose, its properties need to be investigated, to gain understanding of its behavior and usefulness for the problem at hand.

Properties of WIE method - How much more important are k-itemsets compared to k+1-itemsets? As explained, parameter α is the weight of the WIE measure that calibrates the WIE method in calculating the error. It is used in the term ($1/m^{\alpha}$) that reduces weight for larger itemsets. The WIE measure shown in Equation 2 adds errors of all itemsets (size of the largest itemset does not exceed the number of attributes in the data, *m*), which may potentially generate very large number of itemsets (theoretically up to 2^m-1). Because of the ($1/m^{\alpha}$) term, large itemsets contribute very little to the measure since they will be penalized too much. Therefore, it may be possible to execute a faster version of the measure in which only itemsets with supports greater than pre-defined minimum support are used. This can be achieved by executing standard association mining algorithms such as Apriori with different minimum support weights. The upper bound of WIE measure in Equation 2 can be estimated by Euler–Riemann zeta function (Titchmarsh et al. 1986) shown in Equation 3.

Equation 3 Euler–Riemann zeta function

$$C(\alpha) = \sum_{m=1}^{\infty} \frac{1}{m^{\alpha}}$$

The Euler–Riemann zeta function converges for all values of α greater than 1. There is no actual solution for the Euler–Riemann function for odd or decimal values of α greater than 1, but for any positive even integer α=2n, the function is equal to:

Equation 4 Euler–Riemann zeta function for positive even integers

$$\mathrm{C}(2n) = \frac{(-1)^{n+1}\, B_{2n}\, (2\pi)^{2n}}{2(2n)!}$$

where in Equation 4 the B_{2n} is the second Bernoulli number (Carlitz, 1948) which is equal to 1/6.

3.4.1. Behavior of WIE Measure

WIE measure is used as the main evaluation criterion for the binary data created a part of the synthetic claims generation. This includes use in the hyperparameter optimization process (Section 3.5) as a measure to judge the data quality generated by each ML method having a different set of hyperparameters. Parameters of WIE measure, including minimum support of itemset and values of α, affect the behavior of WIE measure (parameter α is the weight of the WIE measure that calibrates the method to measure the error). In this section, WIE measure behavior is evaluated for different values of α (0, 1, 1.4, 3) and different values of minimum support (0, 0.1, 0.005) to find out the best pair of parameters α and minimum support of WIE measure which would make the WIE work in its optimal point for data evaluation.

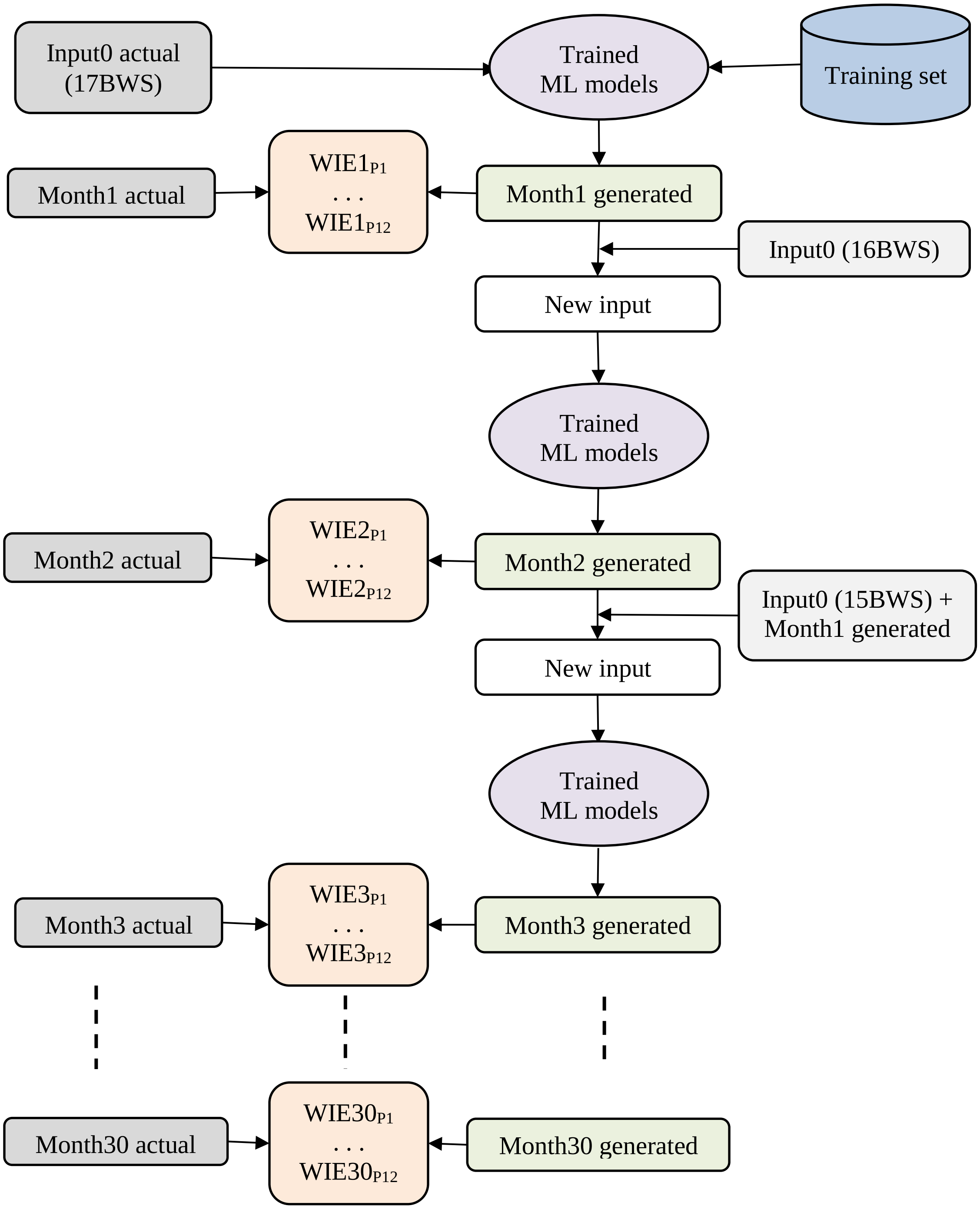

Figure 6 Schema of an experiment for investigating WIE behavior

Figure 6 shows the outline of the experiment conducted in this section where WIE behavior is explored for different minimum support and value of α. The steps in this experiment are as follows:

A. ML models are trained from actual data where back window size is 17 months and forward window size is 1 month. 17 months back window size is selected based on the result of Section 3.3 that showed 17 months back window size is the optimal back window size of ML models.

B. 31 actual datasets (demographic, ELIX and CCS information) are created for a specific cohort (5k patients) where the first actual dataset observes the 5k cohort for a period of 17 months (let's call this input0 dataset). The other 30 datasets observes the same cohort each for one month for the following 30 months (these 30 datasets are named as actual_month1, actual_month2,…, actual_month30). There is a limitation in number of months to be observed because of the large size of the data and the required time that analyses needs to be performed.

C. The first actual dataset (input0) is used as an input of the trained models in Step A and month1 data is generated by the trained models.

D. Using sliding window technique and the same trained models, the last 16 months of input0 dataset merged with the month1 generated data in step C so as to create 17 months of input for the trained ML models in step A. As a result month2 of patient data is generated.

E. Similar to step D and using sliding window technique, the last 15 month of the input0 dataset merged with month1 and month2 generated data (creating 17 month of input again) and used as an input of the trained ML models in Step A to generate month3 data. This process continues until month30 is generated.

F. Once all 30 months is generated, WIE measure with different pair of parameters α and minimum support applied to calculate WIE error between month1 and actual_month1 (this WIE error is named WIE1), between month2 and actul_month2 (WIE2), …, and month30 and actual_month30 (WIE30). Value of α and value of minimum support changes in range 0, 1, 1.4, 3 and 0, 0.1, 0.005 respectively, resulting in 4 graphs which are shown in Figure 7, Figure 8, Figure 9 and Figure 10.

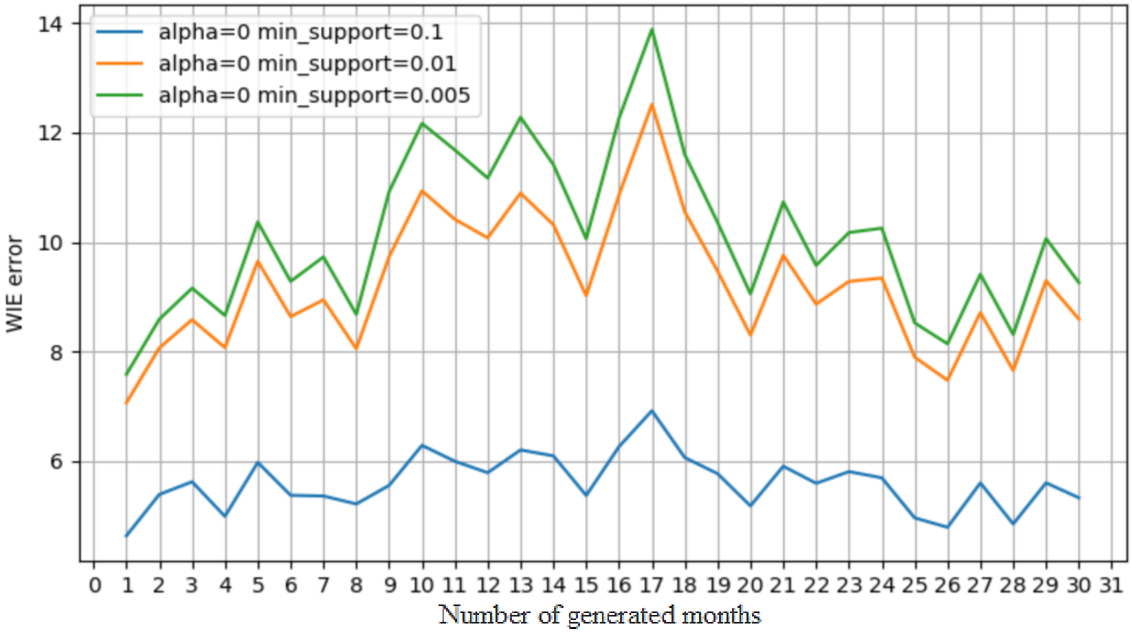

Figure 7 Behavior of WIE measure for α=0 and different minimum supports

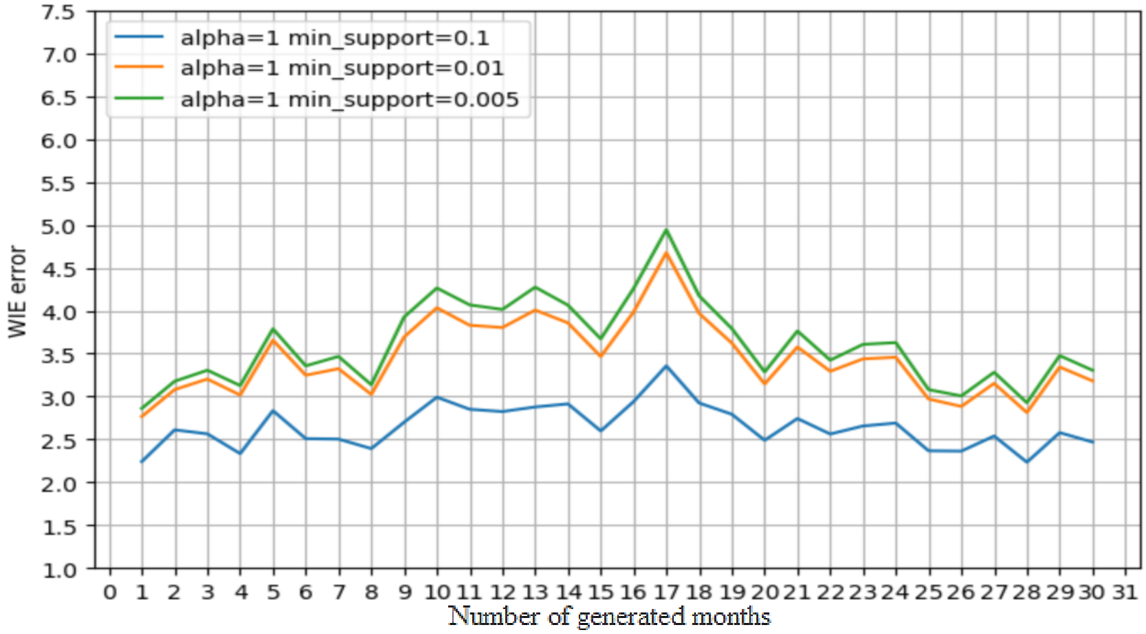

Figure 8 Behavior of WIE measure for α=1 and different minimum supports

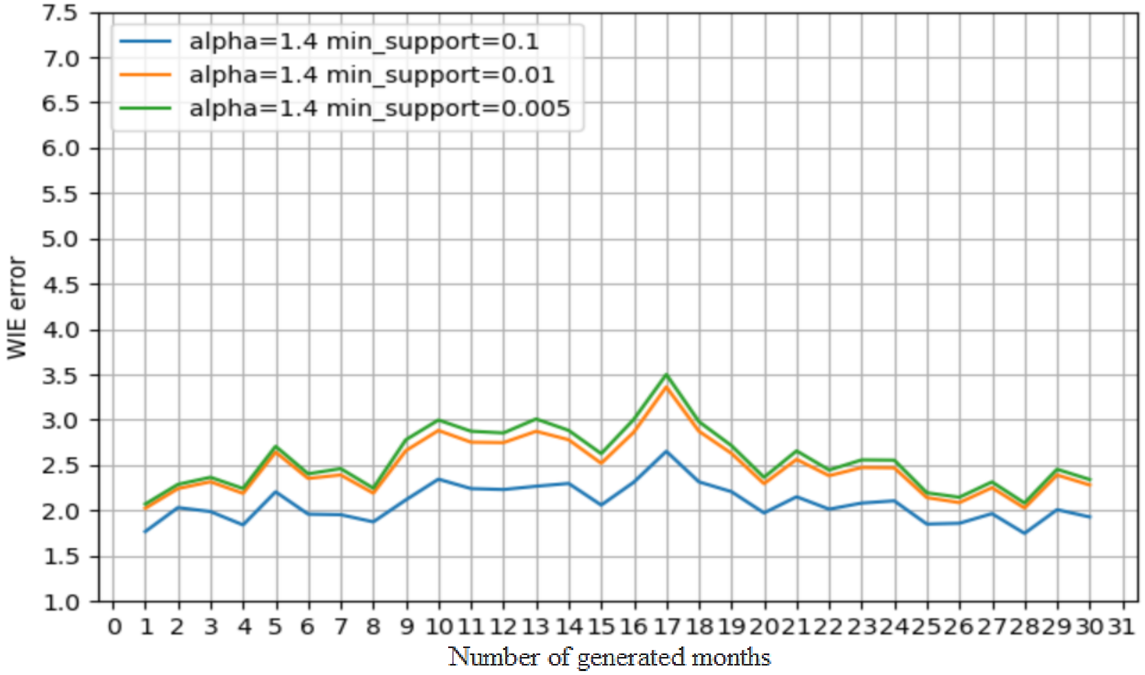

Figure 9 Behavior of WIE measure for α=1.4 and different minimum supports

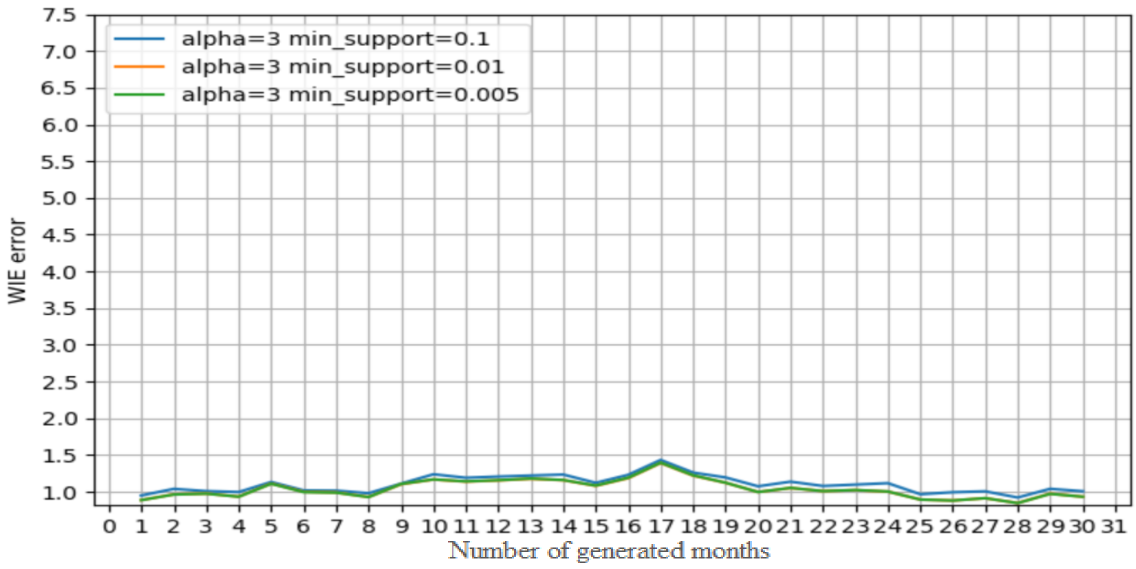

Figure 10 Behavior of WIE measure for α=3 and different minimum supports

It can be noticed from all Figure 7, Figure 8, Figure 9 and Figure 10 that the WIE measure is unstable with minimum support 0.1, which is large in the context of claims data. The reasons for the instability is that there are few itemsets with minimum support 0.1, thus number of itemsets changes by just one (above or below threshold) makes significant difference in the overall WIE error calculation. There seems to be almost no difference in the trend and value of WIE for minimum support of 0.01 and 0.005. In fact, as support decreases more itemsets are compared with each other among the actual and the generated datasets. Hence, smaller support such as 0.01 and 0.005 are better representative for comparing the two datasets. With α = 0, the itemsets with bigger size are boosted too much. In contrast, with α = 3, the itemsets with bigger size are penalized too much. In conclusion, the WIE measure appears to work well with minimum support in the order of 0.01 and α = 1.4, allowing for significant computational gain as compared to generating all possible itemsets of arbitrary size. Therefore, WIE measure is best tuned with minimum support of 0.01 and $\alpha = 1.4$ parameters and WIE is used with these parameters in hyperparameter optimization process (Section 3.5) to evaluate the quality of the data.

3.5. Hyperparameter Optimization of ML Models

In ML field, hyperparameter optimization or hyperparameter tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. There are four main approaches to hyperparameter optimization including Manual Search, Random Search, Grid Search and Bayesian Optimization. In Manual Search, some hyperparameters are chosen based on our judgment/experience and using the chosen hyperparameters a ML model is trained. In Random Search, a grid of hyperparameters is created and a ML model is trained on just some random combination of these hyperparameters. In Grid Search, a grid of hyperparameters is defined and a ML model is trained on each of the possible combinations. Bayesian Optimization builds a probability model of the objective function and uses it to select hyperparameter to evaluate in the true objective function. The objective function is the real-valued function whose value is to be either minimized or maximized over the set of feasible alternatives (Ippolito, 2019; Paul, 2018).

Choosing an appropriate set of hyperparameters is crucial in terms of model performance. In this dissertation, hyperparameter optimization using Grid Search is performed to find out the optimal set of hyperparameters for a ML algorithm that can generate the highest quality patient data. There are three ML algorithm chosen in this experiment including Logistic Regression, Random Forest and Decision Tree. One of the main reason behind choosing these three ML algorithms is that they represent three very different approaches. Decision Tree is a simple and very transparent model. Logistic Regression is a linear model and Random Forest is a highly nonlinear. In addition, since the hyperparameter optimization is a time consuming task so the choice of a reasonably fast ML algorithm in this experiment is matter. Logistic Regression, Random Forest and Decision Tree are typically faster compared with other ML algorithms such as Support Vector Machine and Gradient Boost when they used to be trained on big datasets. The detailed steps of hyperparameter optimization is shown in Figure 11 and explained as follows:

A. ML models with one set of hyperparameter are trained from actual data where back window size is 17 months and forward window size is 1 month. 17 months back window size is selected as optimal back window size based on the result of Section 3.3.

B. 31 actual datasets (demographic, ELIX and CCS information) for a specific cohort (5k patients) are created where the first actual dataset observes the 5k cohort for a period of 17 months (let’s call this input0 dataset). The other 30 datasets observes the same cohort each for one month for the following 30 months (these 30 datasets are named as actual_month1, actual_month2,…, actual_month30). There is a limitation in number of months to be observed because of the large size of the data and the required time that analyses needs to be performed.

C. The first actual dataset (input0) is used as input of the trained models in step A and month1 data is generated by the trained ML models.

D. Using sliding window technique and the same trained models, the last 16 months of input0 dataset merged with the month1 generated data in step C to create 17 months of input for the trained models in step A. As a result, month2 of patient data is generated.

E. Similar to step D and using sliding window technique, the last 15 month of the input0 dataset merged with month1 and month2 generated data (creating 17 month of input again) and used as input of the trained ML models in step A to generate month3 data. This process continues until month30 is generated.

F. The tuned WIE measure (α=1.4 and min support=0.01 found in Section 3.4.1) is used to calculate WIE error between month1 and actual_month1 (This WIE error is named WIE1), between month2 and actul_month2 (WIE2), …, and month30 and actual_month30 (WIE30).

G. Average of WIE1, WIE2, … , WIE30 is calculated (this is named Mean_WIE).

The process above is only for a ML model with one set of hyperparameter. The whole process above (except step B where actual dataset is created only once) is performed for each of the three ML algorithms and each set of their hyperparameters to calculate the mean of WIE error. In the end, the algorithm and its set of hyperparameter that has the lowest mean of WIE is chosen as the model to generate data in Section 4.2.

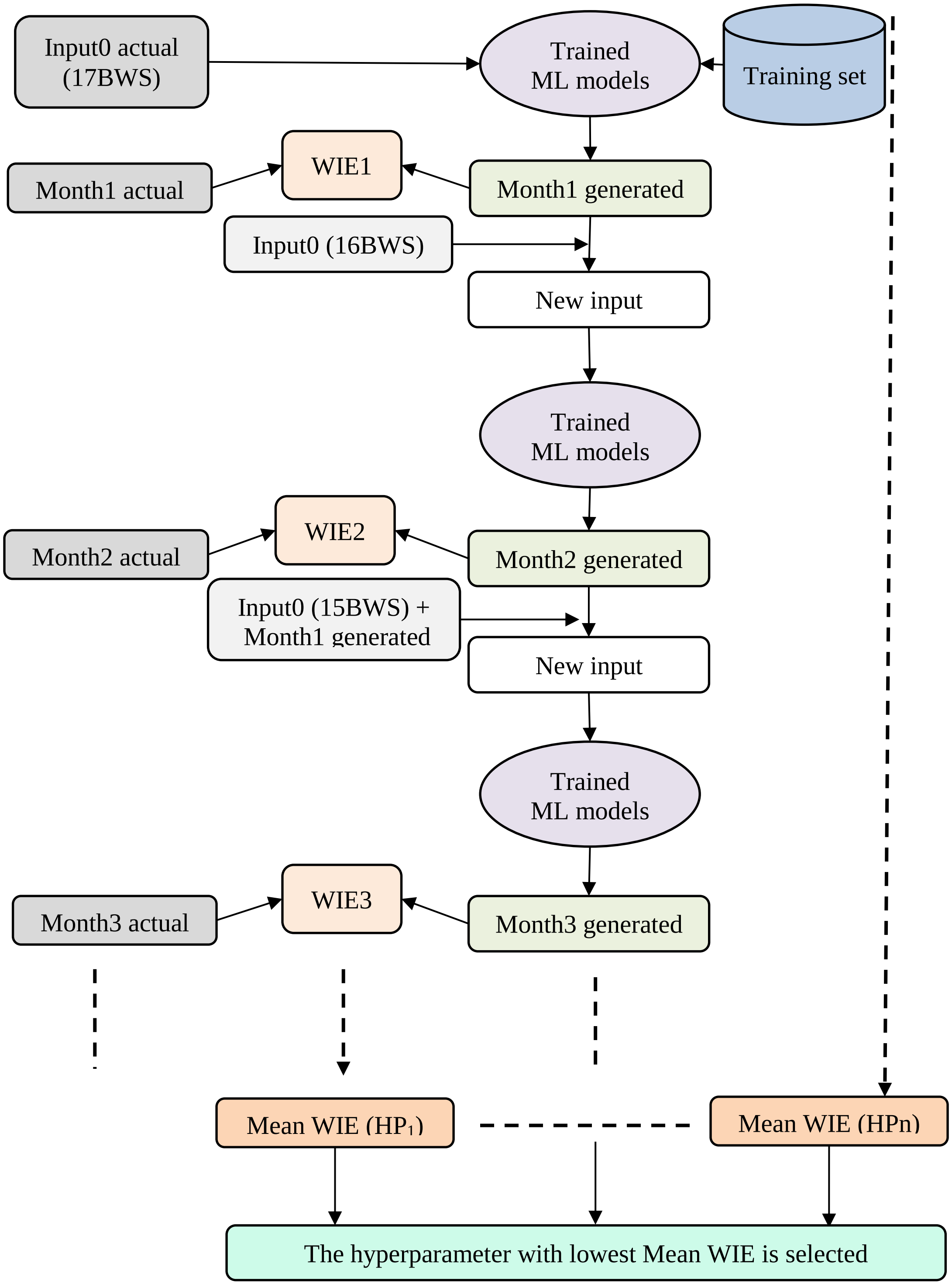

Figure 11 Schema of hyperparameter optimization process for finding the optimal hyperparameter for one ML algorithm

Figure 12, Figure 13 and Figure 14 show hyperparameter optimization results for Decision Tree, Logistic Regression and Random Forest classifiers respectively.

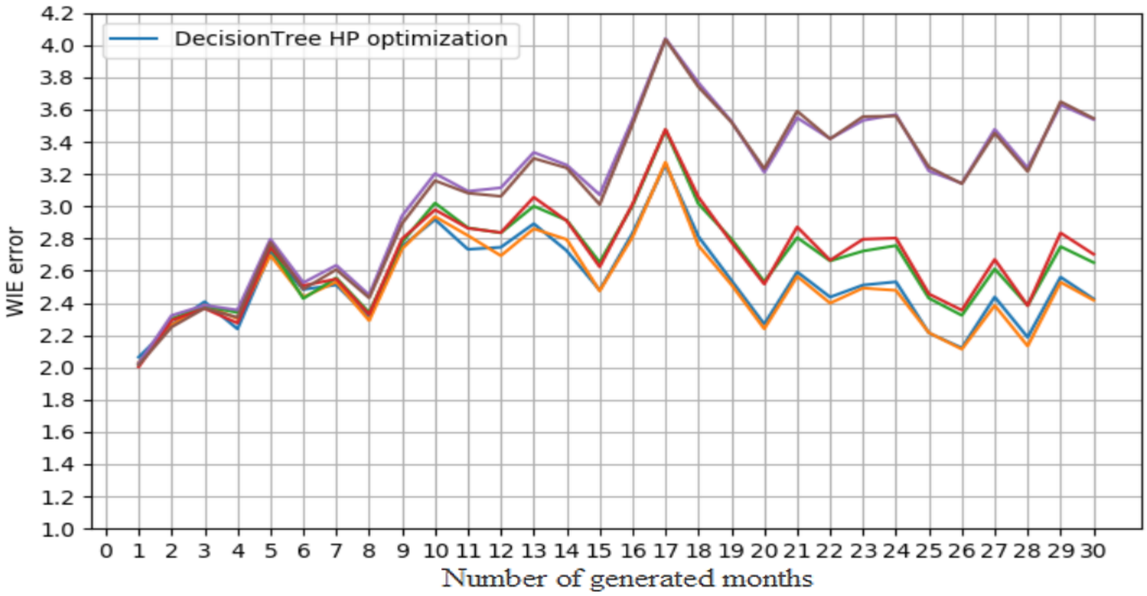

Figure 12 Decision Tree hyperparameter optimization

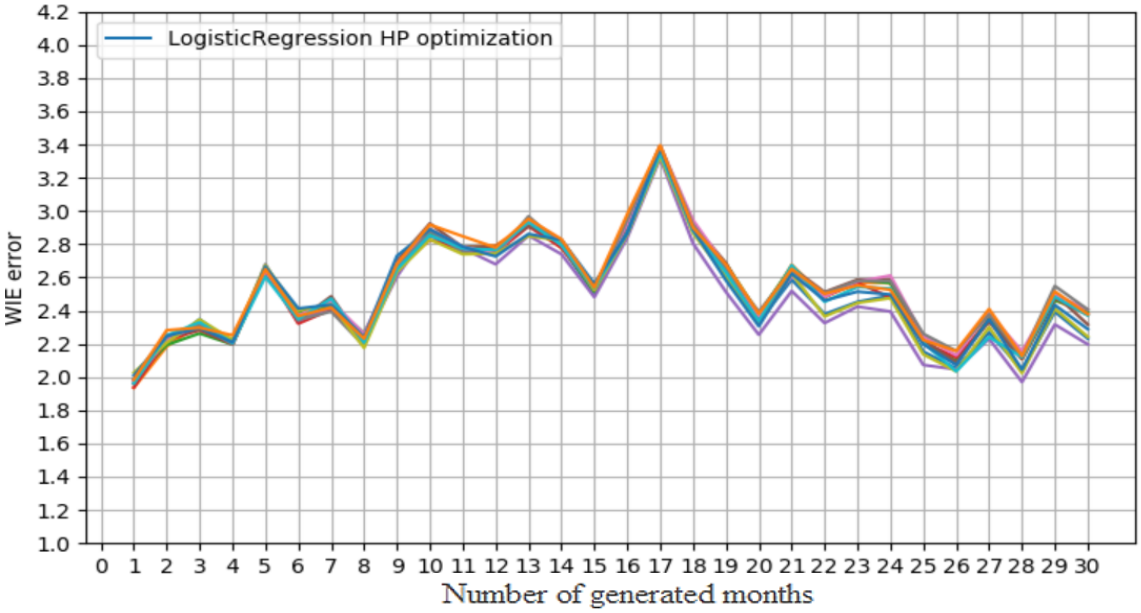

Figure 13 Logistic Regression hyperparameter optimization

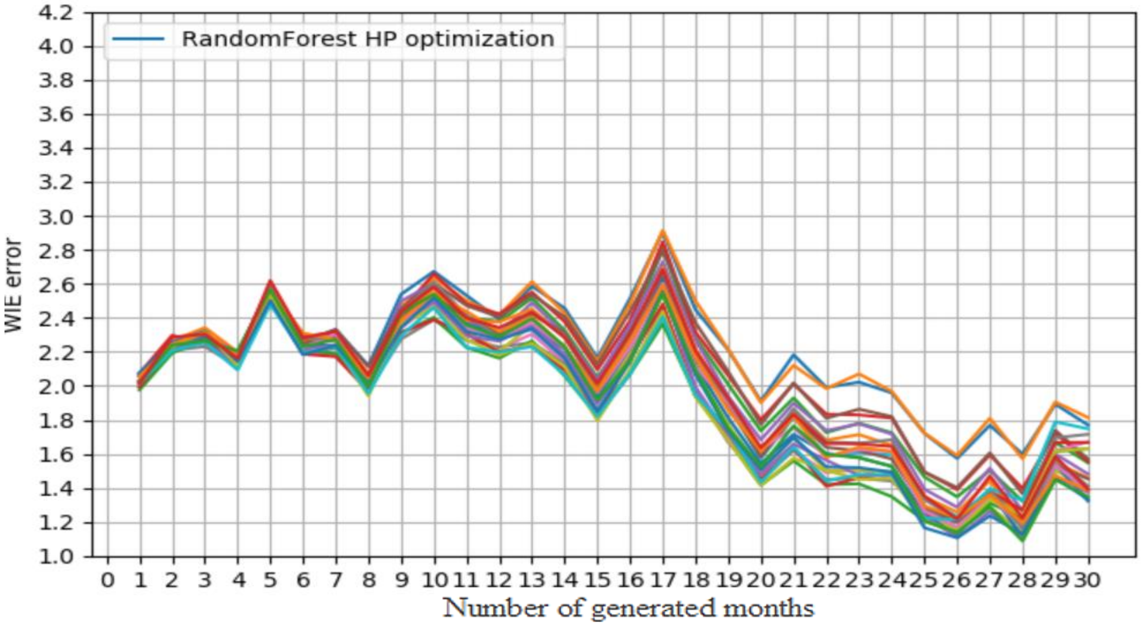

Figure 14 Random Forest hyperparameter optimization

Table 6 Summary of hyperparameter optimization for the three ML algorithms

ML classifier	Minimum mean of WIE	Maximum mean of WIE
Random Forest	1.86 hyperparameter (n_estimators=300, min sample leaf=1,bootstrap=True)	2.19 hyperparameter (n_estimators=10, min sample leaf=1,bootstrap=True)
Logistic Regression	2.45 hyperparameter (penalty=l2, Inverse regularization strength=1, Solver= liblinear)	2.55 hyperparameter (penalty=l2, Inverse regularization strength=1000, Solver= lbfgs)
Decision Tree	2.51 hyperparameter (max depth of the tree=3, criterion= entropy, min sample leaf=6)	3.1 hyperparameter (max depth of the tree=None, criterion= gini, min sample leaf=6) *None: Nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples

Table 6 shows summary of hyperparameter optimization results for the three ML algorithms, Random Forest, Logistic Regression and Decision Tree. According to Table 6 and result of optimization for each algorithm in Figure 12, Figure 13 and Figure 14, it can be seen that there is not a significant difference between values of mean of WIE errors for different hyperparameters within one ML algorithm. However, the difference between values of means of WIE errors across the three ML algorithms are considerable.

The best hyperparameter and ML algorithm that generated the highest quality of data belongs to Random Forest with 1.86 mean of WIE error where its HPs are: n_estimators=300, min sample leaf=1, and bootstrap=True. Here, n_estimators refers to the number of trees built before taking the maximum averages of predictions. Normally, higher number of trees gives a better performance but makes the code to run slower. Min sample leaf is the minimum number of data points allowed in a leaf node. Bootstrap is a method for sampling data points, with or without replacement. Bootstrap has two values, True or False, meaning whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree (Srivastava, 2015).

The Random Forest algorithm with n_estimators =100, min_samples_leaf=1 and bootstrap_val =True has mean of WIE error equal to 1.90. This values is only slightly bigger than WIE error equal to 1.86 which belongs to Random Forest with n_estimators=300, min sample leaf=1, and bootstrap=True. In Section 4.2, the former is chosen for patient data generation, because the difference between the mean of WIE error of the former ML algorithm compared with the latter is very insignificant (0.03), while the former is not as computationally expensive as the latter (due to fewer number of n_estimators).

3.5.1. How Good Is WIE Error of the Tuned ML Model?

In this section, an experiment is conducted to explain how low the value of WIE is for the tuned ML model, Random Forest with n_estimators=300, min sample leaf=1 and bootstrap=True. The WIE error for the tuned ML model is equal to 1.86 obtained in Section 3.5. In this experiment, three set of longitudinal data are generated and WIE applied to measure the WIE error between the longitudinal actual data and each set of the generated longitudinal data. Figure 15 shows the result of this experiment.

In Figure 15, blue line shows WIE error between 30-month actual datasets and 30-month generated data with all values of zeros. As can be seen from the figure, the mean of WIE error for the dataset with all values of zeros is 6.2. The green line in Figure 15 shows WIE error between 30-month actual datasets and 30-month data generated by Logistic Regression with not tuned hyperparameter. The mean of WIE error for the 30-months data generated by a not tuned Logistic Regression drops from 6.2 to 2.5. The purple line shows WIE error between 30-month actual dataset and 30-month data generated by the tuned hyperparameter algorithm which is Random Forest with n_estimators=300, min sample leaf=1, and bootstrap=True. The mean of WIE error for the 30-months data generated by the tuned hyperparameter algorithm drops from 6.2 to 1.87.

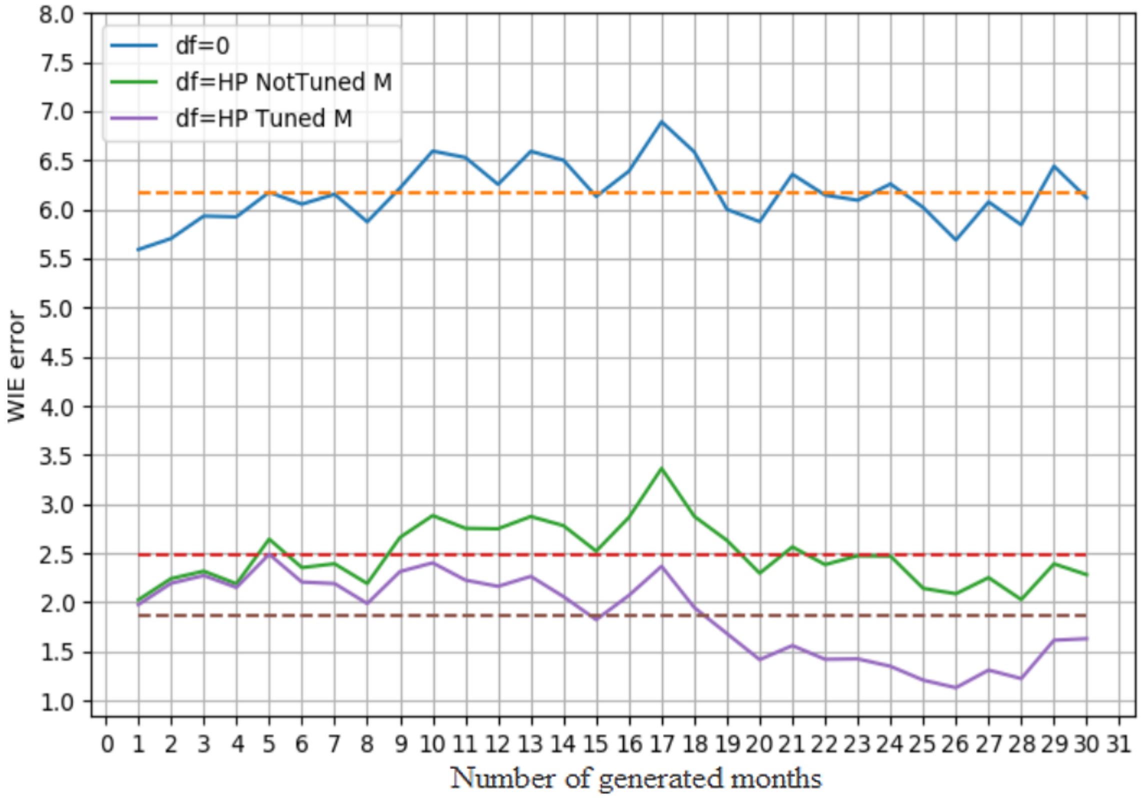

Figure 15 WIE error between actual data and three generated datasets

A couple of important points can be obtained from Figure 15:

- Generated dataset with all values of zeros has no itemsets and so no itemsets would match with itemsets in the actual data when WIE is used to compare the generated data against actual data. That’s the reason WIE error jumps high for the dataset with all values of zeros (blue line) and so the mean of WIE for this dataset is as high as 6.2. Itemsets in the data generated by ML algorithms (green and purple lines) matches with a considerable range of different itemsets in the actual data, and that’s the reason the green and purple lines are considerably lower compared with the blue line.

- Comparing the generated data using ML algorithms with each other (green and purple lines), it can be noticed that WIE error for data generated by the tuned-hyperparameter ML models (purple line) has a decline trend as more months are generated. This indicates the quality of the data generated by the tuned-hyperparameter ML models as more and more monthly data are generated.

3.5.1.1. Question: Why the Shape of All Lines in Figure 15 Follows a Similar Pattern?

To understand the reason why the shape of all lines in Figure 15 follows a similar pattern, an experiment is conducted in this section.

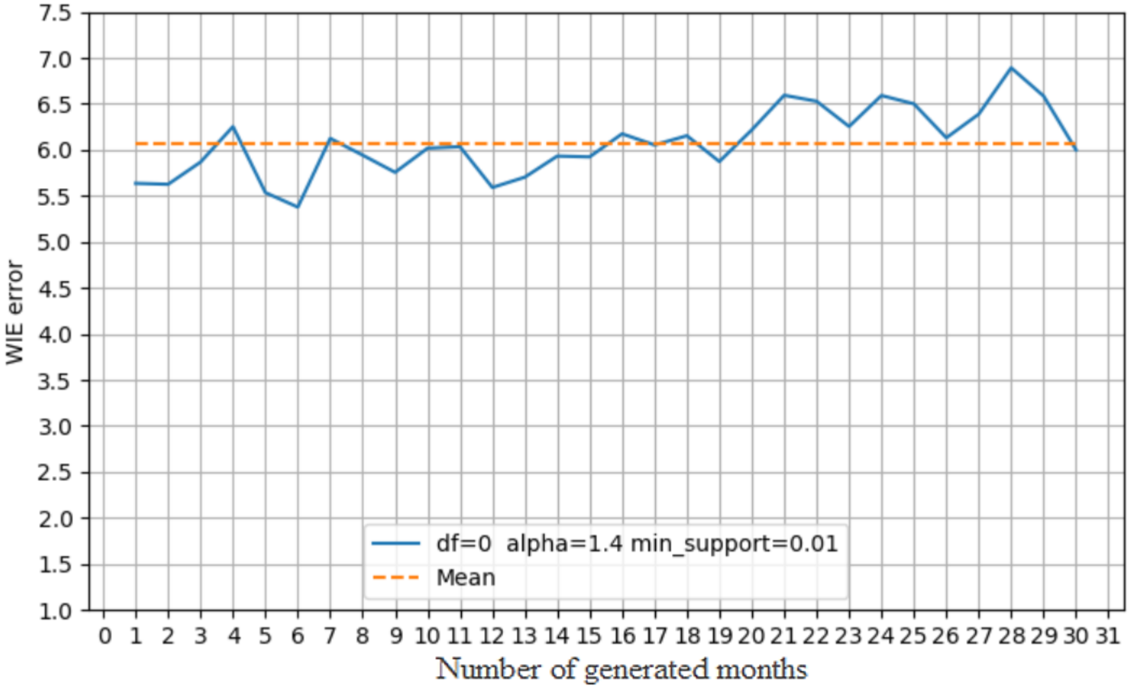

Figure 16 WIE error between a different set of actual data and all zero generated dataset

The blue line in Figure 16 shows WIE error between 30-month actual dataset and 30-month dataset with all values of zeros. The 30-month actual data used in this experiment belongs to the same cohort used in the experiment in Section 3.5.1, but it observes the cohort for a different time period. The time period has shifted 330 days back or in other words month1 starts from day 180 in this experiment versus the time period in the experiment conducted in Section 3.5.1 starts from day 510. As can be seen from Figure 16 and blue line in Figure 15, a different trend/pattern in WIE error can be noticed. The difference between the trend/pattern in WIE error among these blue lines is solely related to the property of the two datasets which means frequency of 0s and 1s and distribution of 0s and 1s are different in the two datasets. Since the two datasets are different, as a result, values of itemsets supports are different for the two datasets and this makes the tuned WIE measures to show different values of errors when compares each of these two datasets with a dataset having values of all zeros.

CHAPTER FOUR: DATA GENERATION METHODS

Chapter Four contains eleven sections. In section 4.1, initial data used for starting data generation process is explained. Data generation methods and generating patient level data are discussed in Section 4.2. Section 4.3 explains generating death date for patients and Section 4.4 presents an approach for converting patient level data into claim level data. The remaining sections 4.5 – 4.11 provide details of the process of generating claim attributes including claim number, claim through date, ICD10, HCPCS, provider specialty, and NPI.

4.1. Generating Initial Data: Census Demographics

Simulated census demographic data is used as the initial input of ML models for generating monthly patient level data explained in Section 4.2.1 (See Figure 18). Simulated census demographic data is generated based on methods described by Wojtusiak (2016) where it consists of the following 4 steps as shown in Figure 17.

A. Gender of a patient is generated using probabilistic models and based on the distribution of sex in Census demographic data.

B. Date of birth of a patient is generated using probabilistic models and based on distribution of age for each gender. Age is restricted to 65 and above for the purpose of this dissertation.

C. Race of a patient is generated using probabilistic models and based on his/her gender in step A.

D. State of a patient is generated using probabilistic models and based on distribution of race in each state.

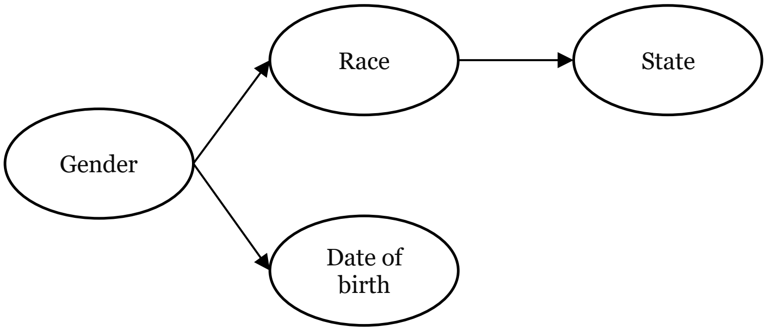

Figure 17 Process of generating simulated demographic information

The four steps above are repeated/looped for as many patients as one would desire to generate. The distribution information of sex with age, sex with race, and race with state are based on demographic information of general US population as indicated by statistics from Social Security Administration and US Census Bureau (Census, 2020). Demographic information of patients needs to be generated only for month 1 and that will be used as demographic information for the entire time that claims data are generated for patients. This is because date of birth, race, gender and location of patients will be the same no matter how many years of data are generated for patients. It is assumed that no patient will change their state during the period of time that data are generated.

4.2. Generating Patient Level Data

From the results of Section 3.3 and Section 3.5, it was concluded that in order to generate the highest quality monthly patient level data, three main features have to be implemented in model construction. These three features are: The optimal back window size (17 months), training set size of 30k unique patients, and Random Forest algorithm with hyperparameter: n_estimators=100, min sample leaf=1, bootstrap=True.

4.2.1. Generate Data with the Same Size as Optimal Back Window

Simulated data with the same size as optimal back window (17 months is the optimal back window size obtained in Section 3.3) need to be generated first in order to be used as input of the most tuned data generation models. To generate the optimal back window data, the following steps have to be performed as shown in Figure 18.

Step 1- month1: Models are trained (shown as M_d in the figure) using actual data where the independent variables are demographics (sex, race and age) and dependent variables are each of diagnosis (ELIX) and procedures (CCS) attributes observed for one month. Next, the simulated demographic data created in Section 4.1 are used as the input of these trained models and one-month data with diagnosis and procedure attributes are generated. The data generated in this step (demographic, diagnoses and procedures) are named as month1 data.

Step 2-month2: Another set of models are trained (shown as M_1 in the figure) where independent variables are demographic, diagnosis and procedure attributes observed for one month and dependent variables are each of diagnosis and procedure attributes observed for its following month. The month1 data created in step 1 is used as input of these models and as a result, the following month of data (all diagnosis and procedure attributes) are generated. The generated data in this step are named as month2 data.

Step 3- month3: Another set of models are trained (shown as M_2 in the figure) same as models trained in step 1, but, with the only difference that back window size is 2 months for these models (independent variables are observed for two months). Month1 and month2 data generated in step 1 and step 2 are combined/merged and used as the input of these models to generate the following month data. The data generated in this step are named as month3 data.

Step 4- Remaining months: This process continues until data same size as the optimal back window size are created (it continues until month17 data are created).

The optimal back window data created in these four steps are used as the first input in data generation method in Section 4.2.2 to initialize data generation. Figure 18 shows the process of generating data with the same size as optimal back window.

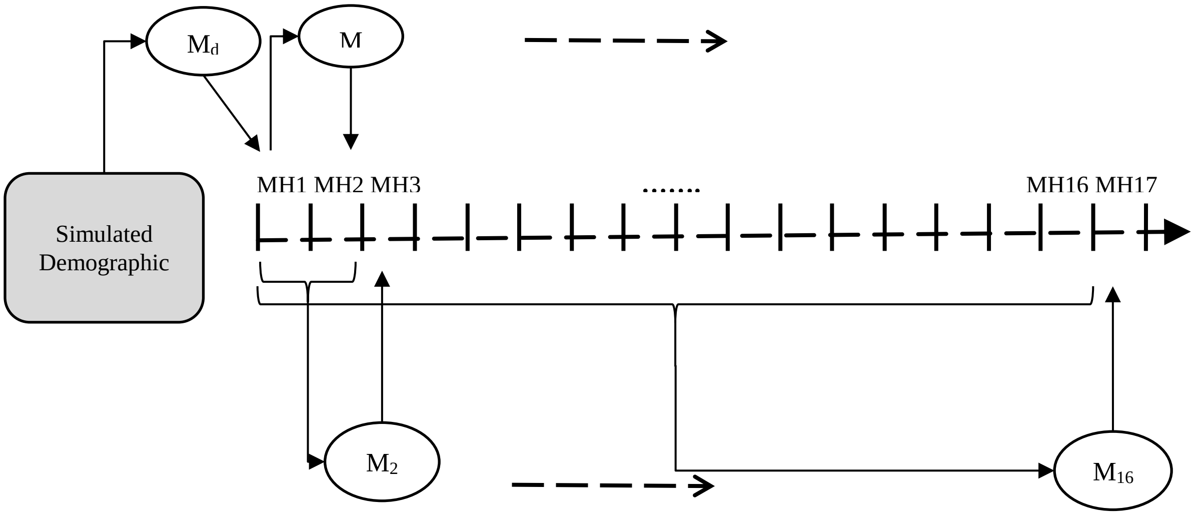

Figure 18 Simplified schema of generating data with the same size as optimal back window

4.2.2. Generating Monthly Patient Level Data

There are three main steps for generating monthly patient level data explained as follows.

Step 1- The data generated in Section 4.2.1 (the 17 months data) is used as the initial data of the tuned ML models (Random Forest algorithm with n_estimators=100, min sample leaf=1, bootstrap=True obtained in Section 3.5) so as to generate the following month of data. The data generated in this step is the first month desirable generated data.

Step 2- Sliding window technique is used and shifted one month forward. Therefore, the last 16 months of the initial data in step 1 and 1-month data generated in step 1 are merged and used as input of the tuned ML models and as a result second month desirable data are generated.

Step 3- This process of sliding window is continued and monthly patient data are generated iteratively. This can continue as much data as it is desired to generate or until the last patient die.

Figure 19 shows the process of generating monthly patient level data.

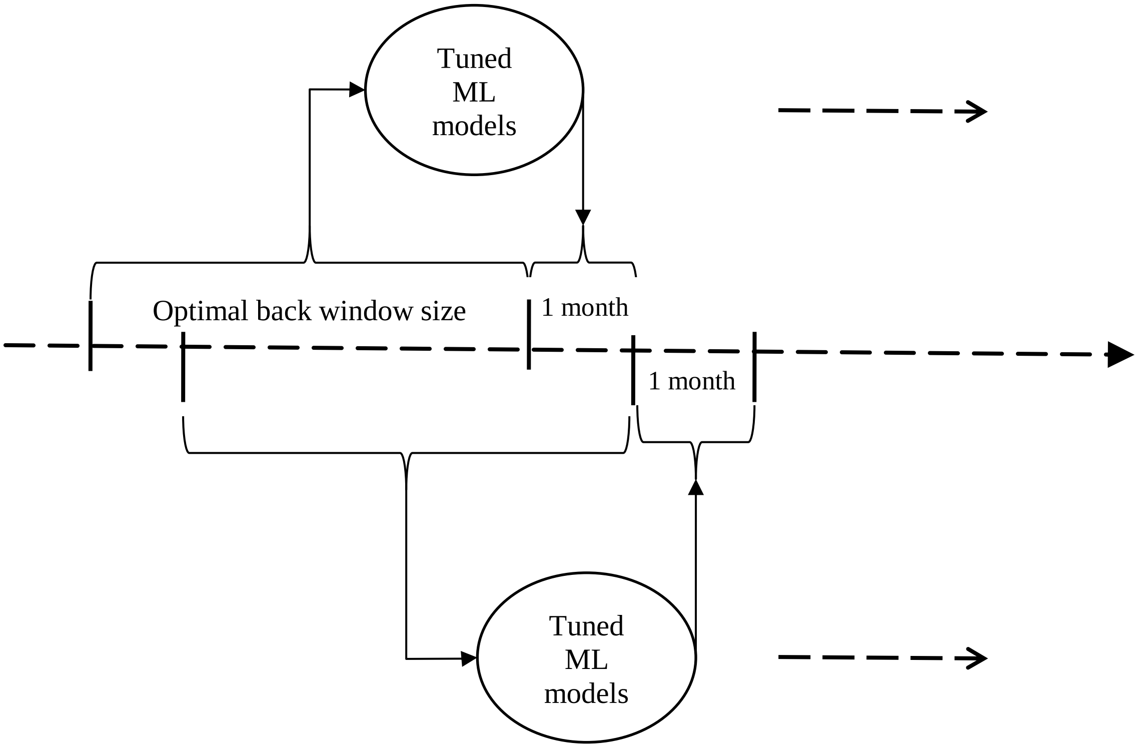

Figure 19 Generating monthly patient level data

4.3. Predicting Date of Death

In the previously described method for generating data in which diagnoses and procedures are predicted based on previous time window, synthetic patients "live" forever since there is no mechanism to stop generating data after some point. This section describes a mechanism to generate date of death for patients from which date data will not be generated for patients.

A classification model is trained using the actual patient data where independent variables are demographic, all diagnoses and procedures (ELIX and CCS attributes observed for a month) and outcome is a binary variable showing whether a patient will die in the following month. Next, each time data are generated for a certain month (Section 4.2.2), the generated data are used as the seed of the classification model trained in this section to predict whether a patient will die in the following month. If a patient is predicted to die in the following month, a random number within the date range of the following month is selected for the death date of that patient. Table 7 shows a simplified version of generated patient level data.

Table 7 Generated patient level data (a simplified format)

patid_gen*[1]	demographic	claim date	ELIX_i*[3] (max)	ELIX_j (max)	ELIX_k (max)	ELIX_p (max)	…	remaining ELIX (max)
100	ASRD*[2]	month1	1	0	0	1	…	0
101	ASRD	month1	1	0	0	0		0
102	ASRD	month1	0	0	0	1		0
103	ASRD	month1	1	0	1	1		0

*[1] Patient ID for generated data
*[2] ASRD: Age, Sex, Race, Death date
*[3] i, j, k and p can be any number from 1 to 30

4.4. Generating Claim Level Data

So far, patient level data are generated. The next step is to convert the patient level data, shown in simplified format in Table 7, into claim level data. ML models together with probabilistic models are applied to generate monthly diagnoses for claim base table. The following four main steps needs to be performed in order to convert patient level data to claim level data.

1. Predict total number of claims in a given month for generated patients.
2. Predict count of each diagnoses (ELIX) for generated patients.
3. Use association rules to calculate supports of diagnosis itemsets from actual data.
4. Generate claim level data and assign diagnoses to each claim based on information obtained from the first three steps.

4.4.1. Predicting Count of Claims for Each Patient

In order to convert patient level data to claim level data, the first question that needs to be answered is how many claims each patient should have? To answer this question, we can predict count of claims for each generated patient. In order to predict count of claims for each patient in generated patient level data, a training set, similar to Table 8 is created from actual data. Variables in Table 8 are demographic, maximum of each diagnoses (ELIX) in a month, maximum of each procedures (CCS) in a month and number of claims in the same month for a patient. Table 8 which is patient level data is used to train a regression model where count of claims is the outcome variable and the remaining variables in this table are independent variables.

Table 8 A schema of the training set for predicting count of claims. Created from actual data

patid*	demographic	max of each ELIX in a month	max of each CCS in a month	number of claims (in the same month)
1501	ASR			13
1502	ASR			14
1503	ASR			10
1504	ASR			9

patid*:patient ID

The monthly-generated patient level data has all the independent variables in Table 8 and so they will be used as input of the trained regression model to predict count of claims for each generated patient. Table 9 shows the result for one generated patient.

Table 9 Predicted count of claims for one generated patient

patid_gen	demographic	claim date	$ELIX_i$ (max)	$ELIX_j$ (max)	$ELIX_k$ (max)	$ELIX_p$ (max)	predicted count of claims (from step 1)
103	ASR	month1	1	0	1	1	6

By now, we have the count of claims for each generated patient. The next question is what diagnosis (ELIX) codes should be assigned to each generated claims? Having the count of claims and information about maximum of each diagnosis (ELIX) (shown in Table 9) for a generated patient is not helpful enough to assign diagnosis codes to each generated claims. This can be seen in Table 10, where there are different ways that values of $ELIX_i$, $ELIX_k$ and $ELIX_p$ can be assigned to claims for patid_gen=103. Not all of these diagnosis assignments are necessarily correct from claims perspective. For example, imagine $ELIX_i$ is related to eye surgery and $ELIX_k$ is related to foot surgery. Hence, it is not reasonable that $ELIX_i$ and $ELIX_k$ shows up in one claim. This makes us to do the next two steps, which will eventually enable us to assign values of diagnoses to each generated claim reasonably.

Table 10 Claim level data for one generated patient

patid_gen	claim_no*	demographic	claim date	$ELIX_i$ (max)	$ELIX_k$ (max)	$ELIX_p$ (max)
103	1	ASR	*5*	*0?*	*0?*	*1?*
103	2	ASR	*5*	*0?*	*1?*	*1?*
103	3	ASR	*5*	*1?*	*0?*	*0?*
103	4	ASR	*7*	*1?*	*0?*	*1?*
103	5	ASR	*9*	*1?*	*1?*	*0?*
103	6	ASR	*10*	*1?*	*1?*	*1?*

claim_no*: claim number

4.4.2. Predicting Count of Each Diagnosis for Generated Patients

In order to assign diagnoses to each generated claim reasonably, in this section, count of each diagnosis attribute is predicted for a generated patient. Similar to Table 8, which was used for predicting number of claims for generated patients, Table 11 is created from actual data and is used to train 30 regression models. Each of the regression model predicts count of each diagnosis codes for each generated patient in the same month. The independent variables of each of the 30 regression models is demographic information, maximum of each diagnoses (ELIX), maximum of each procedures (CCS), and claims count. Claims count obtained in Section 4.4.1.

Table 11 Schema of training set for predicting count of diagnoses. Created from actual data

patid	demo	max of each ELIX in a month	Max of each CCS in a month	count of claims in a month	count of $ELIX_i$	count of $ELIX_j$	count of $ELIX_k$
1501	ASR			13	13	3	2
1502	ASR			14	3	2	0
1503	ASR			10	3	0	1
1504	ASR			9	1	0	0

Two important notes need to be mentioned in this section:

First note: In order to train a regression model to predict count of $ELIX_i$, only patients having count of $ELIX_i$ more than 0 are used in training set for predicting count of $ELIX_i$. This would improve the regression model predicting $ELIX_{i,}$ since unnecessary patients are dropped. The same logic applies for predicting count of other diagnosis attributes.

Second note: Count of $ELIX_i$ is only predicted for generated patients that has value of $ELIX_i$_max=1. It is not reasonable to predict count of $ELIX_i$ for generated patients having value of $ELIX_i$_max =0. The same logic applies to other diagnosis attributes.

Table 12 shows predicted count of diagnoses for one generated patient (patid_gen=103).

Table 12 Predicted count of diagnoses for one generated patient

patid_gen	demo	claim date	$ELIX_i$ (max)	$ELIX_j$ (max)	$ELIX_k$ (max)	$ELIX_p$ (max)	count of claims	predicted count of $ELIX_i$	predicted count of $ELIX_k$	predicted count of $ELIX_p$
103	ASR	month 1	1	0	1	1	6	3	2	2

4.4.3. Calculating Supports of Diagnosis Itemsets from Actual Data

Having the information about number of claims and number of diagnoses for each generated patient obtained from the last two steps are not sufficient yet to assign diagnoses to each claim. We need one more piece of information which shows us the chance that two or more than two diagnosis codes can show up in a claim. To calculate the chance that two or more than two diagnosis codes can show up together, association rules are used and applied to the actual Medicare claim table to find out the supports values of each diagnosis as well as support values for pairs of diagnoses. There are 30 diagnosis attributes (29 ELIX and “other”) and hence 435 (30*29/2) unique combinations pairs of diagnoses. Itemset supports of single diagnosis and pair of diagnoses attributes enable us to assign diagnoses to each claim reasonably. The process of how diagnoses are assigned to each generated claim is explained in Section 4.4.4.

4.4.4. Generating Claim Level Data and Assign Diagnoses to Each Claim

From the previous three sections (Section 4.4.1, Section 4.4.2, and Section 4.4.3), three information are obtained: 1) count of claims for generated patients, 2) count of diagnoses for generated patients, and 3) support of single diagnosis and diagnoses pairs. Suppose for a generated patient, information presented in Table 13 are obtained (patid_gen=103).

Table 13 Predicted count of claims and diagnoses for one generated patient

patid_gen	predicted count of claims	predicted count of $ELIX_i$	predicted count of $ELIX_p$	predicted count of $ELIX_k$
103	6	3	2	3

Based on the information obtained from the three previous sections (Section 4.4.1, Section 4.4.2, and Section 4.4.3), claims are generated and diagnoses (ELIXs) are assigned to each claim. The approach for generating claims and assigning diagnoses is explained for one patient as follows and the same logic can be generalized to generate claims and assign diagnoses to all generated patients.

Notice the example showed in Table 13 for one generated patient (patid_gen=103). To generate the 6 claims for patid_gen=103, first, one observation having $ELIX_i$ as 1 is generated and this observation is considered as one claim. Since predicted count of $ELIX_i$ is 3, second observation having $ELIX_i$ as 1 is generated, but from the second observation it is checked every time whether it is still allowed to add the observation as a claim to the previous generated claims. This is checked according to the predicted number of claims for the generated patient. There are two situations because of checking this constraint:

1- Number of generated claims for the patient is smaller than his/her predicted number of claims and so it is still allowed to add the next observation to his/her generated claims. In this situation, two probabilities are calculated. A) Merging probability: The probability that the observation can be merged with patient' previous claims. B) Appending probability: The probability that the observation can be added as a new claim to patient' previous claims (calculating merging probability and appending probability of an observation is explained in Section 4.4.4.1). Having the merging and appending probability, the next observation is either merged or appended with patient' previous claims.

2- Number of generated claims of the patient is equal with his/her predicted number of claims and so from this point it is not allowed to add the next observation to his/her previous claims. In this situation, only merging probability is calculated and the observation is assigned based on merging probability (the appending probability is zero).

4.4.4.1. Calculating Probability of Merge and Append for an Observation

In this section, calculating the merging probability of an observation with previous claims of a patient as well as appending probability of an observation to previous claims of a patient are explained.

Probability of merge: As a reminder, each observation has only a single diagnosis attribute (one ELIX attribute). To calculate the probability that an observation can be merged with other diagnosis attributes of a single claim, the probability of intersection of diagnosis attributes needs to be calculated. The probability of intersection of n events is shown in Equation 5.

Equation 5 Intersection probability of n events

$P\ (A_1 \cap A_2 \cap A_3 \cap\ \dots\ \cap A_n)$

where in this case A_n is a diagnosis ($ELIX_n$) and n can change from 1 to 30 since there are 29 ELIX and “others” variables. The probability of intersection for two diagnosis is the supports of diagnosis pairs showing with each other. Supports of diagnosis pairs are calculated in Section 4.4.3 and they are used in this section as well (30*29/2 = 435 diagnosis pairs).

To calculate the intersection probability for more than two diagnoses is computationally expensive because of the number of combinations. There are 30 diagnosis attributes and so the number of combinations are $\frac{30!}{30} = 29! = 8.841762e + 30$ (an integer with 31 numbers). Therefore, to be able to calculate the intersection probability for more than two diagnoses, it is assumed that diagnosis (ELIX) attributes are independent. Equation 6 shows the probability of intersection for n independent events.

Equation 6 Intersection probability of n independent events

$P(A_1 \cap A_2 \cap A_3 \cap \dots \cap A_n) = P(A_1)*p(A_2)*P(A_3)*\ \dots .\ *\ P(A_n)$

Probability of append: The appending probability of an observation to previous claims of a patient is shown in Equation 7. *n* is the number of previously assigned claims for a patient.

Equation 7 Probability of append

Probability of append= $1-\sum_{n=0}^{n}(\text{merging probability of the observation with a single claim of the patient})$

4.5. Generating Claim Through Date and Claim Number for Claim Table

Claim through date is the last day on the billing statement covering services provided to a patient. Claim through date is randomly generated for each claim within the days of a month of generated data. However, if a patient is predicted to die in that month, claim through dates for the claims of that patient are randomly generated within the beginning of that month until the patient' death date. For example, if a patient is not predicted to die in the first month, days between 1 and 30 is generated randomly as claims through date of that patient. In the case that the patient is predicted to die in day 20 in the first month for example, claim through date for claims of that patient is randomly generated within 1 to 20. It is assumed that a month has 30 days.

Claim number is a unique identification number assigned to each claim in a claim table. To generate claim number, first, monthly-generated claim data is sorted based on claim through date. Next, a unique number as claim number is assigned to each claim based on the order of the claim through date. See Table 14.

Table 14 Generating claim through date and claim number

patid_gen	all diagnoses	Sorted claim through date*	claim number
101		1	1
120		2	2
127		3	3
127		3	4
103		4	5
102		4	6

*Date is shown based on days of a month

4.6. Converting ELIX to ICD10 in Claim Table

So far, diagnosis attributes in the form of ELIX codes are generated for claim table. ELIX codes were used to group ICD10 of patients to decrease the complexity of analysis and improve efficiency of prediction models. In the actual Medicare claims data diagnosis codes are shown as ICD9 or ICD10 (ICD10 became effective from October 2015 onwards).

The simplest approach to convert each ELIX to ICD10 in the generated claim table is to randomly choose one of the ICD10 code that belongs to that ELIX group and replace it with that ELIX. This is not a very accurate and a robust approach. The approach taken in this dissertation for converting ELIX to ICD10 is to use distribution probability of ICD10 in each group of ELIX found in the actual claims data and convert ELIX to ICD10 based on that probability. A simplified version of distribution probability of ICD10 in each group of ELIX based on actual claims data is shown in Table 15.

To create Table 15, first, distribution of each ICD10 is calculated from actual claims data. Next, ICD10 are grouped by their related ELIX (the distribution of all ICD10 in each group of ELIX adds up to 1). Table 15 is shown in a simplified format for the sake of explanation. The actual table has couple of thousands of rows.

Table 15 A schema of distribution probability of ICD10 in each group of ELIX

ELIX	ICD10	Distribution
8	icd1	0.3
8	icd2	0.5
8	icd3	0.2
7	icd4	0.4
7	icd5	0.1
7	icd6	0.5

4.7. Generating Primary Diagnosis Code for Claim Table

In the actual Medicare claim table, each claim has a primary diagnosis code which is the reason for admission/encounter/visit of a patient and it is chiefly responsible for the services (such as procedures) provided to the patient (CMS, 2019). Since each claim in the generated claim table has more than one diagnosis codes most of the times, hence one of the diagnosis code in each generated claim has to be chosen as the primary diagnosis code. One approach to do this is to choose one ICD10 from all ICD10 in each generated claim randomly. However, this is not a robust approach.

To do that in this dissertation, first, the probability of distribution of each primary diagnosis code is calculated from the actual claim data as shown in Table 16 (a simplified version). Next, based on information in Table 16, an ICD10 among all the ICD10 in each claim is chosen as primary ICD10 for that claim. Table 16 is shown in a simplified format for the sake of explanation. The actual table has couple of thousands of rows.

Table 16 A schema of probability of ICD10 to be primary diagnosis code

Diagnosis code	Probability
icd1	0.0001
icd2	0.0003
icd3	0.0004
icd4	0.0001

The great point about this approach is that if an ICD10 can't be a primary diagnosis for a claim, probability of that ICD10 from actual data would be 0 or very low in Table 16 and so that ICD10 would be never chosen as a primary diagnosis in the generated claim table. At this point, claim table looks like Table 17 (primary diagnosis is shown as dgns1). The only remaining variable that needs to be added to claims table is referring NPI which is explained in Section 4.11.

Table 17 A schema of generated claim table

patid_gen	demographic	claim_no	Claim date	dgns1	dgns2	dgns3	dgns4	…	dgns12
1	ASRD*	1	date1	icd1	null	null	null	null	null
1	ASRD	2	date2	icd2	icd3	null	null	null	null

*ASRD: Age, Sex, Race, Death Date

4.8. Creating Line Table and Assigning HCPCS Codes to Each Line

HCPCS codes show the procedures performed on patients because of their primary diagnosis condition. Each claim in the claim table can have at least one procedure code (it is a claim so it has at least one procedure code). Therefore, a new table, named “line table” should be created where each claim of the claim table has one or multiple lines in the line table showing all the procedures performed on a patient for that claim. In order to create line table, two questions need to be answered:

Question1: How many lines (or HCPCS codes) should be generated for each claim (each primary diagnosis) located in claim table?

Question2: What HCPCS code should be assigned to each line created for that claim?

To answer question 1, count of lines for each primary diagnosis (each claim has one primary diagnosis in the line table) of claims is calculated from actual line table. As a result, a table similar to Table 18 is obtained. Table 18 is shown in a simplified format for the sake of explanation. The actual table has millions of rows.

Table 18 A schema of count of lines for each primary diagnosis; obtained from actual line table

primary diagnosis	count of lines
icd10_a	2
icd10_a	1
icd10_a	6
icd10_b	1
icd10_b	3

Next, mean and standard deviation of count lines for each primary diagnosis showed in Table 18 are calculated. As a result, a table similar to Table 19 is obtained. Table 19 is shown in a simplified format for the sake of explanation. The actual table has couple of thousands of rows (same size as all the ICD10 codes); 50 observations of the actual table can be seen in Table 36 under Appendix.

Table 19 A Schema of mean and standard deviation of count lines for each primary diagnosis; obtained from actual line table

primary diagnosis	mean of count lines	standard deviation of count lines
icd10_a	3	2.65
icd10_b	2	1.41

Next, count of claims for each primary diagnosis is calculated from generated claim table (Table 17 shows generated claim table). As a result, a table similar to Table 20 is obtained. Table 20 is shown in a simplified format for the sake of explanation. The size of actual table can vary depends on the number of generated patients.

Table 20 A schema of count of claims for each primary diagnosis; obtained from generated claim table

primary diagnosis	count of all claims
icd10_a	10
icd10_b	45000

Using the mean, standard deviation of each primary diagnosis obtained from actual line table (Table 19) and count of each primary diagnosis obtained from generated claim table (Table 20), normal distributed samples are created for each primary diagnosis in the generated line table. For example, according to Table 20, the sample size for primary diagnosis Icd10_a has 10 different values, and based on Table 19, mean of count lines and standard deviation of count lines for primary diagnosis Icd10_a is 3 and 2.65 respectively. Therefore, the created sample showing number of lines for primary diagnosis icd10_a in generated line table is a list similar to list in Equation 8.

Equation 8 Sample showing number of lines for a primary diagnosis code in generated line table

s_icd10_a=[3,6,5,2,3,5,3,5,1,2]

where s_icd10_a means sample of lines for primary diagnosis icd10_a.

Each value in Equation 8 indicates number of lines to be created for primary diagnosis icd10_a in generated line table. For example, 3 means one of the 10 claims in generated claim table that have icd10_a as their primary diagnosis should have 3 lines. The same logic applies to the other values in Equation 8. It is assumed that the distribution of count lines of each primary diagnosis is a normal distribution.

Just a note that the values in the normal distributed generated samples are not necessarily a positive integer (it can happen to be negative or not integer sometimes). Hence, all float numbers are rounded up to their closet integers and negative and 0 values are changed to 1 since the number of procedure lines for each claim must be a positive integer (at least 1).

Another solution to answer question 1 is to train a regression model using Table 18 as well as demographic and other related variables of patients to predict count of lines for each primary diagnosis code in the generated line table. This method is, however, computationally very expensive because primary diagnosis should be converted into dummy variables and since there are about 70,000 ICD10 codes, so the training set size should be big enough to obtain a reasonably low mean squared error for the regression model.

Question2: What HCPCS code should be assigned to each line created for that claim?

First, it is worth to remind that each claim has one primary diagnosis code. To answer question 2, the probability that each HCPCS code can show up with primary diagnosis codes is calculated from actual line table. To explain this with an example, suppose Table 21 is obtained from actual line table, which shows probability that each primary diagnosis can show up with HCPCS codes (Table 21 is shown in a simplified format for the sake of explanation. The actual table can have thousands HCPCS for each primary diagnosis code; 50 observations of the actual table can be seen in Table 37 under Appendix).

Suppose in the generated claim table, the primary diagnosis for a claim is icd10_a, and the number of predicted procedure lines for that claim from Question 1 is two. Therefore, based on the probability shown for diagnosis icd10_a in Table 21, 2 of the procedure codes are chosen for those two lines.

Table 21 A schema of probability that each primary diagnosis can show up with HCPCS codes; obtained from actual line table

primary diagnosis	HCPCS	probability
icd10_a	hcpcs_a	0.1
icd10_a	hcpcs_b	0.5
icd10_a	hcpcs_c	0.3
icd10_a	hcpcs_d	0.1

4.9. Generating Performing Provider Specialty Code for Line Table

In Medicare data, specialty code is used in the line table for pricing the line item service assigned by the Medicare Administrative Contractor (MAC) based on the corresponding NPI number (CMS, 2019). Table 38 in Appendix shows all the specialty/service provider codes in Medicare data. From the analysis of actual line table for about 50 million patients' claims, 0.999 of the claims have one specialty code. Hence, it is assumed that each claim has one specialty code and therefore, in this dissertation, one specialty code is assigned for all lines of a generated claim in generated line table.

Specialty/service code of performing physicians/service providers are directly related to the procedures that they perform on patients. Hence, in order to generate specialty code for a claim in generated line table, procedures codes generated in Section 4.8 are used. In details, the distribution probability of each procedure codes (HCPCS) in relation to specialty codes is calculated from the actual line table. Next, based on these probabilities, specialty codes are assigned. Table 22 shows a schema of probability distribution of procedures codes in relation to specialty codes (for performing physician). Table 22 is shown in a simplified format for the sake of explanation. The actual table has couple of thousands of observations. Based on the probability in Table 22, one specialty code is chosen for a procedure code in generated line table. Since, there are normally multiple HCPCS codes in one claim, this ends up having multiple specialty code for a claim in most of the cases. Because one claim should have one specialty code, the specialty code that shows up the most for a claim is chosen. If multiple specialty codes for a claim have the same frequency of showing up, one is chosen randomly.

Table 22 A schema of procedures codes and performing physician specialty codes probability distribution

HCPCS	specialty code	probability
hcpcs_a	spec_a	0.1
hcpcs_a	spec_b	0.5
hcpcs_a	spec_c	0.2
hcpcs_a	spec_d	0.2
hcpcs_b	spec_a	0.4
hcpcs_b	spec_c	0.2
hcpcs_b	spec_e	0.4

4.10. Generating Performing NPI for Line Table

A National Provider Identifier (NPI) is a unique identification number issued to health care providers in the United States by CMS. Performing NPI refers to a physician, surgeon, or lab officer who performs a procedure on a patient. In this dissertation, in order to generate Performing NPI (P_NPI) two main information from actual line table need to be obtained.

A. Distribution of service providers' population in terms of each specialty from actual line table (Table 23 is shown in a simplified format for the sake of explanation. The actual table can be seen under Appendix in Table 39).

B. Ratio of number of P_NPI to the number of patients from actual data. In one year data (and only 5 percent of claims) from actual line table, the number of P_NPI is

1,035,055 and number of patients is 1,811,122 and so the ratio is: 1,035,055 /1,811,122=0.57

Table 23 A schema of percent of service providers' population for each specialty from actual line table

specialty	percent of P_NPI for each specialty
spec_a	0.2
spec_b	0.1
spec_c	0.7

Number of physicians/service providers in each specialty has a positive relation with the number of patients. Based on the result in A and B, and having the number of generated patients for each state, Table 24 can be achieved for generated patients (number of generated patients is assumed to be 0.5 Million for VA and 1 Million for FL in Table 24). Table 24 shows the number of P_NPI for each specialty in each state for generated line table. It is assumed that each state has the same distribution of service providers in regards to the specialties.

Table 24 Number of P_NPI for each specialty in each state for generated line table

state	specialty	percent of P_NPI for each specialty from actual data	number of generated patients in each state	ratio of P_NPI to patients in actual data	number of P_NPI for each state for generated data	number of P_NPI for each specialty in each state for generated data
VA	spec_a	0.2	0.5M	0.57	0.5M*0.57	0.2*0.5M*0.57
	spec_b	0.1	0.5M	0.57	0.5M*0.57	0.1*0.5M*0.57
	spec_c	0.7	0.5M	0.57	0.5M*0.57	0.7*0.5M*0.57
FL	spec_a	0.2	1M	0.57	1M*0.57	0.2*1M*0.57
	spec_b	0.1	1M	0.57	1M*0.57	0.1*1M*0.57
	spec_c	0.7	1M	0.57	1M*0.57	0.7*1M*0.57

Table 25 can be simply generated once we have information of Table 24. Table 25 shows a schema of P_NPI table for generated line table.

Table 25 A schema of P_NPI table for generated line table

P_NPI	specialty	state
10000001	spec_a	VA
10000002	spec_a	VA
10000003	spec_a	VA
10000004	spec_a	VA
10000005	spec_b	VA
10000006	spec_b	VA
10000007	spec_b	VA

10000008	spec_b	VA
….	…	…
10040000	spec_a	FL
10040001	spec_a	FL
10040002	spec_a	FL

To assign P_NPI to a claim, the specialty code in that claim and the place of living of the patient are matched with specialty code and place of living of P_NPI and accordingly one P_NPI is assigned to a claim in generated line table.

One may argue that why patients are assigned to P_NPI using states and why not counties or zip codes. To explain this, patients and P_NPI are assigned using states because in the real world situation, it happens quite frequently that a patient who seek procedures travels to other county and zip codes since most of the times a specialist for those procedures is not located in the same zip code or county that the patient resides. Therefore, it is a reasonable assumption that patients who seek procedures move within a state and their movement is not necessarily limited within the county or the zip code they reside. However, the most precise approach is to perform network analysis and find out patients' movement trend form actual data when they seek treatments. At this point, generated line table is complete and it looks like Table 26.

Table 26 A schema of generated line table

patid_gen	demographic	claim_no	claim date	line_no	primary diagnosis	HCPCS	specialty	P_NPI
100	ASR	1	date_a	1	icd10_a	hcpcs_a	spec_a	15001
100	ASR	1	date_a	2	icd10_a	hcpcs_b	spec_a	15001
100	ASR	1	date_a	3	icd10_a	hcpcs_c	spec_a	15001

4.11. Generating Referring NPI for Claim Table

A referring physician is one who requests an item or service for a patient. In actual Medicare claims data, about 40 percent of the claims have different NPI in the claim table, or in other words, about 40 percent of the claims' R_NPI (Referring NPI) are different with P_NPI. In order to assign R_NPI to generated claim table, two questions needs to be answered first.

Question 1: Which claims have R_NPI same with P_NPI?

Question 2: What to assign as R_NPI to those claims that have R_NPI different with P_NPI?

In order to answer these two question, Table 27 is obtained from actual claim tables where it shows specialties and rate of claims having different NPI for each specialty. Table 27 shows only a couple of rows of the actual table. The actual table is shown in Table 40 under Appendix. For example, service code 69, and 63, where all of their claims have almost 100 percent chance to have different NPI in claim tables, are Clinical laboratory (billing independently) and Portable X-Ray Supplier (Billing Independently).

Table 27 A schema of specialty and rate of claims having different NPI for each specialty; obtained from actual claim tables

specialty/service code	rate of claims having different NPI for each specialty
69	0.99
63	0.99
44	0.54
21	0.53

Since each claim in generated line table has only one specialty code, rate of claims having different NPI for each specialty in Table 27 is used to assign the same P_NPI to R_NPI or a different NPI for R_NPI. If the R_NPI is not the same as P_NPI, a new NPI needs to be assigned for that claim in generated claim table. Since, the actual Medicare claim table has no specialty, therefore, one simple approach is to assign a random number as R_NPI for those claims that have different NPI with P_NPI.

However, to do it reasonably, 100 NPI ID is randomly chosen from actual Medicare claim table and almost all of them have MD as specialty. Therefore, it is assumed that all those claims in generated claim table that have different NPI have MD as specialty. This enables us to assign provider to patient claims by specialty and state code (the same way that it was performed for P_NPI in Section 4.10). As a result, a R_NPI table similar to Table 25 needs to be created for generated claim table so as to be used for assigning R_NPI to only those claims whose R_NPI is different with P_NPI. To create the R_NPI table, similar to what was done in Section 4.10 for creating P_NPI table, the number of patients having different NPI in the actual line table (1,603,229) as well as the number of R_NPI for the claims with different NPI (809,950) in actual line table are calculated. Ratio of number of NPI to patients is 0.5. On the other hand, number of patients in generated claim table that does not have R_NPI is calculated (let us say X; depends on the size of generated patient, X can change). Knowing X and ratio of number of providers to number of patients from actual claim table (0.5) enables us to calculate the number of R_NPI for generated claim table and accordingly creating Table 28. Table 28 shows a schema of R_NPI table for the generated claim table. R_NPI are assigned to the claims whose R_NPI is different with P_NPI based on states of patients and states of referring providers.

Table 28 A schema of R_NPI table for the generated claim table

R_NPI	state
15000001	VA
15000002	VA
15000003	VA
15000004	VA
15000005	VA
15000006	VA
15000007	VA
15000008	VA
….	…
15040000	NC
15040001	NC
15040002	NC

CHAPTER FIVE: RESULTS OF DATA GENERATION

In this chapter, results of generation of multiple sets of patient data are described. The presented method has been applied to construct numerous datasets, with notable ones described here. Additionally, 100 observations of the simulated demographic table can be seen in Table 33 under Appendix. 100 observations of generated claim table can be seen in Table 34 under Appendix. 100 observations of generated line table can be seen in Table 35 under Appendix. The validity of generated data was previously evaluated by WIE measure. In order to add other layers of validation to the generated data, in Section 5.1, summary statistics of the 20 generated datasets are compared with that of Medicare data. Besides, in Section 5.2, two datasets are generated and the death rate of the generated datasets are compared with that of Medicare data.

5.1. Summary Statistics of the Generated Data vs. Medicare Data

In order to add another layer of validation to the generated data, statistical information of the generated datasets are summarized and compared with that of actual Medicare claims data. Statistical information of one year of Medicare claims data (for 5% of patients) are summarized and presented in Table 29 (demographic) and Table 30 (claim tables).

Table 29 Summary statistics of Medicare demographic table (5% of patients)

patient count	mean age	most common race	count of the most common race	most common sex	count of the most common sex	most common state	count of the most common state
1,811,122	70.3	1 (white)	1,482,595	2 (female)	1,018,009	05 (California)	153,652

Table 30 Summary statistics of Medicare claim tables (5% of patients in carrier)

claims count	mean claims count - patient year	unique count of p_dgns	most common p_dgns	count of the most common p_dgns	unique count of hcpcs	most common hcpcs	count of the most common hcpcs	unique count of specialty	most common specialty	count of the most common specialty
45,673,594	25.2	36,077	I10	1,760,215	11,459	99214	5,191,890	93	69	13,779,832

* p_dgns: primary diagnosis

Table 31 Summary statistics of demographic information for 20 generated datasets

set	patient count	mean age	most common race	count of the most common race	most common sex	count of the most common sex	most common state	count of the most common state
1	100	74.8	1	66	1	53	05	13
2	100	75.7	1	62	2	52	05	10
3	100	75.2	1	56	1	52	05	16
4	100	74.1	1	58	1	52	05	12
5	100	74.6	1	64	2	51	05	13
6	100	75.3	1	66	2	59	05	15
7	100	75.7	1	69	1	53	05	10
8	100	75.7	1	63	2	55	45	8

9	100	74.9	1	65	1	59	10	14
10	100	74.2	1	67	1	54	05	11
1	1000	75.3	1	637	2	518	05	103
2	1000	75.2	1	645	2	532	05	125
3	1000	75.1	1	662	1	514	05	119
4	1000	75.1	1	635	1	520	05	126
5	1000	74.8	1	627	1	528	05	123
6	1000	75.2	1	655	2	503	05	127
7	1000	75.3	1	624	2	507	05	101
8	1000	75.3	1	631	1	530	05	105
9	1000	74.9	1	635	2	507	05	116
10	1000	74.7	1	617	1	508	05	128

20 sets of patient claims data are generated and their statistical information are summarized as shown in Table 31 and Table 32. In each set, two years of longitudinal data are generated starting from 1, January, 2010. As explained in Section 4.1, demographic information are generated based on statistical information of population-level from Social Security Administration and US Census Bureau (Census, 2020). According to Table 31, average patients' age changes between 74 and 76 for the 20 generated datasets. The most common race among generated datasets is white (1) in all the 20 generated datasets. Most common gender is not fixed and it changes between male (1) and female (2) among the 20 generated datasets. Most common state is California (05) for all the generated datasets except for two of them where the most common states is Florida (10) and Texas (45).

Demographic information of generated data are based on Census data and it cannot be really compared with demographic information of Medicate data since they are two different population. Nevertheless, some common features can be still noticed among the demographic of generated datasets and demographic of Medicare data as presented in Table 29 and Table 31. For example, White race and California are the most common race and state in all generated datasets and Medicare claim tables.

Table 32 Summary statistics of claims information for 20 generated datasets

set	patient count	claims count	mean claims count-patient year	unique count of p_dgns	most common p_dgns	count of the most common p_dgns	unique count of hcpcs	most common hcpcs	count of the most common hcpcs	unique count of specialty	most common specialty	count of the most common specialty
1	100	3396	17.8	364	I10	637	755	99214	625	70	69	2802
2	100	3774	19.6	376	I10	689	775	99214	754	69	69	3223
3	100	3427	18.7	373	I10	625	739	99214	617	69	69	2845
4	100	3637	19.1	386	I10	651	762	99214	703	69	69	3136
5	100	3293	17	357	I10	653	714	99214	607	72	69	2875
6	100	3871	20	397	I10	715	768	99214	704	74	69	3208
7	100	3402	18	374	I10	657	771	99214	627	67	69	2786
8	100	2866	15.5	338	I10	559	699	99214	526	69	69	2479
9	100	3101	16.5	352	I10	560	700	99214	641	68	69	2495
10	100	3404	18.2	409	I10	575	773	99214	618	71	69	2854
1	1000	36456	19.1	957	I10	6690	1865	99214	6909	84	69	30543
2	1000	35035	18.4	927	I10	6527	1792	99214	6649	84	69	29520
3	1000	33744	17.7	914	I10	6254	1752	99214	6543	85	69	28476
4	1000	33268	17.8	914	I10	6318	1806	99214	6360	84	69	27897
5	1000	33561	17.7	931	I10	6600	1839	99214	6362	85	69	28060
6	1000	33127	17.5	910	I10	6185	1827	99214	6212	83	69	27995
7	1000	34530	18.2	884	I10	6502	1806	99214	6560	85	69	28886
8	1000	34226	18	939	I10	6487	1830	99214	6469	88	69	28544

9	1000	34489	18.1	920	I10	6490	1774	99214	6503	85	69	29238
10	1000	34562	18.3	919	I10	6446	1801	99214	6736	84	69	28845

* p_dgns: primary diagnosis

Table 32 shows the summary statistics of claims information for the 20 generated datasets. According to this table, number of claims for cohorts of 100 patients changes between 2866 and 3871, and it changes between 33,268 and 36,456 for cohorts of 1000 patients. The average number of claims for each generated patient in a year changes between 15.5 and 20. The most common primary diagnosis code is Hypertension (I10) and the most common HCPCS code is “office or other outpatient visit” (99214) among the 20 generated datasets. The most common service provider codes or specialty is clinical laboratory (69) among the 20 generated datasets.

By comparing summary statistics of Medicare claims data (Table 30) and claims information for the 20 generated datasets (Table 32), it can be seen that the most common primary diagnosis code between the generated data and the actual data is Hypertension (I10). Similarly, most common procedure code is office or other outpatient visit (99214) and most common Specialty/Service Provider code is clinical laboratory (69) in both generated and actual datasets. The generated datasets are stored in GMU serves, mli10.

5.2. Death Rate in the Generated Data vs Medicare Data

In order to add another layer of validation to the generated data, two sets of 10 years longitudinal datasets are generated, starting from year 2010, each for 100 patients. Next, the average percentage of death of each generated dataset are compared with that of

Medicare data. As can be seen from Figure 20 and Figure 21, the number of patients survived after 10 years is 29 in the first generated dataset and 23 for the second generated dataset. The average percentage of death in a year for patients shown in Figure 20 is 7.1 and it is 7.7 for the patients shown in Figure 21. This number is close to that of actual Medicare data in which average percentage of death in a year is 8.

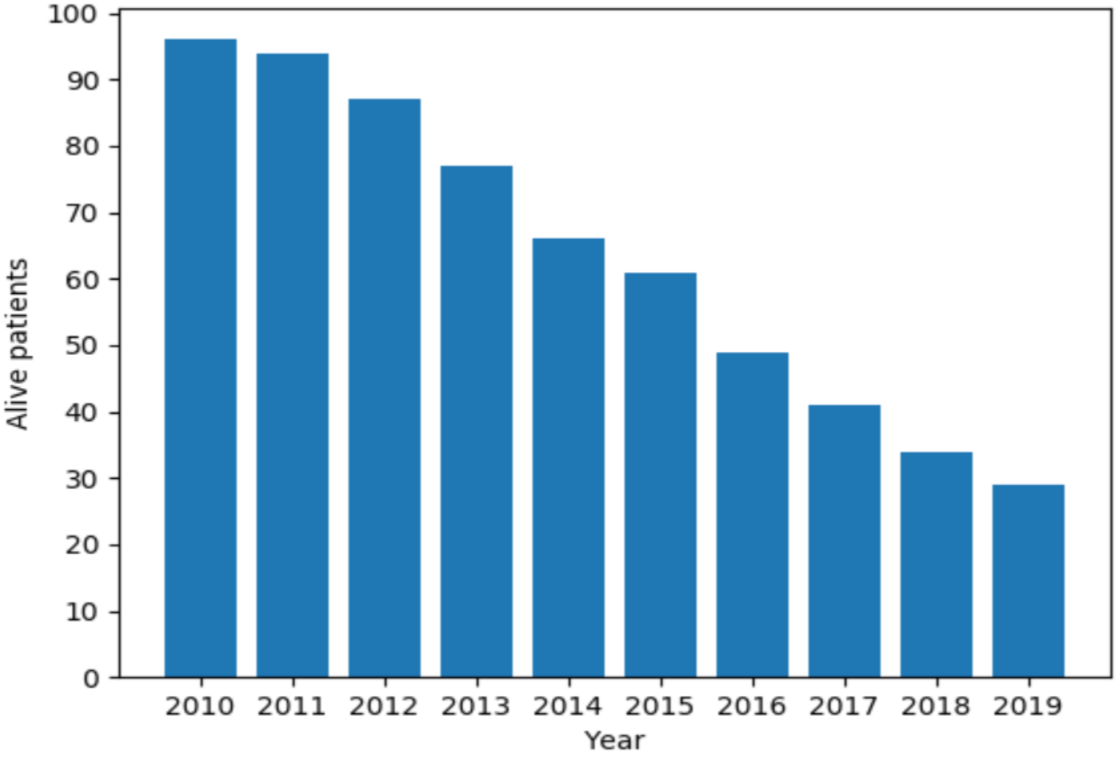

Figure 20 Survival rate in the generated data (Dataset 1)

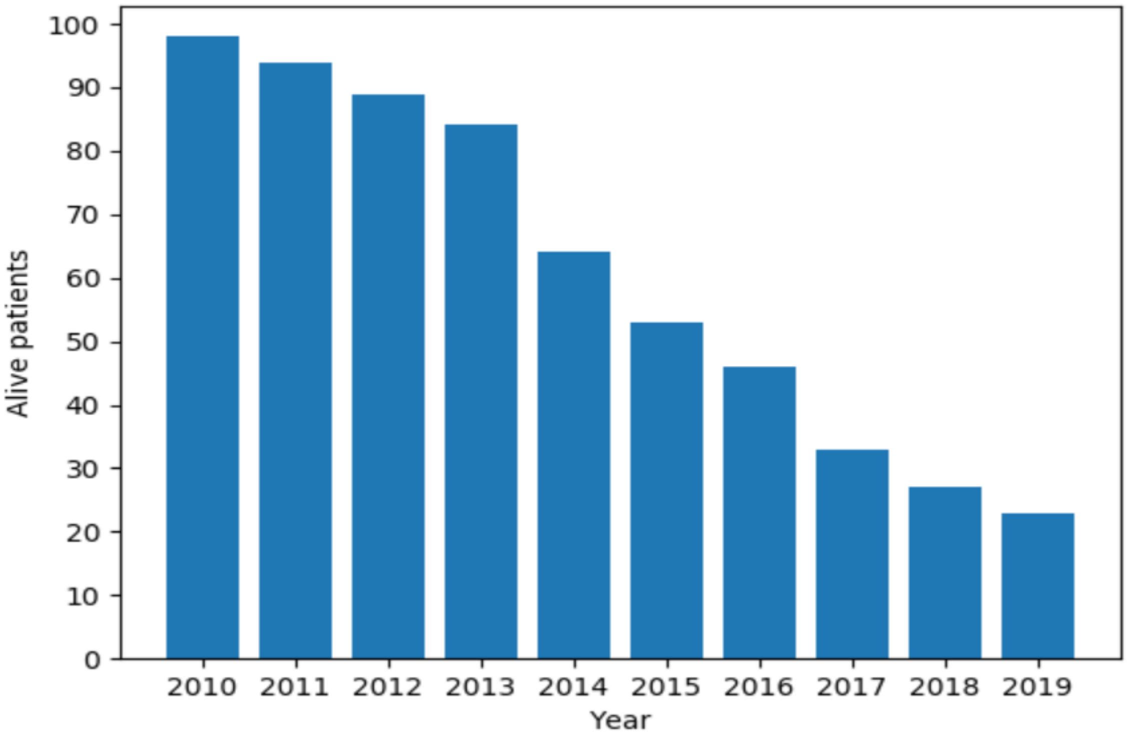

Figure 21 Survival rate in the generated data (Dataset 2)

CHAPTER SIX: CONTRIBUTION AND CONCLUSION

Chapter 6 includes three main sections. In Section 6.1, contributions of the dissertation are listed. Limitations and future works are discussed in Section 6.2. Finally, conclusion is presented in Section 6.3.

6.1. Contribution of the Dissertation

The main contributions of the dissertation are as follows:

A comprehensive literature on synthetic data generation in healthcare domain: In this dissertation, a comprehensive literature on data generator methods, specifically in healthcare domain is presented. This will help readers and practitioners in effectively adopting data generator approaches and provides an insight into its state-of-the-art.

New ML-based method for generating synthetic patient claims data: ML models can be used to predict future and to generate data iteratively. In this dissertation, longitudinal patient claims data are created using ML models. To construct a good model for generating highest quality claims data, two important steps are performed. First, using sliding window techniques, the back window size of ML models for generating patient data is investigated and the optimal back window size is chosen. Second, hyperparameter optimization of ML algorithms is performed and the most tuned algorithm is selected. The developed method can be applied to create longitudinal patient claims data not limited to any specific disease. It can also be applied to generate as many datasets as one desire with any sizes.

New approach for synthetic data evaluation: Previous research mainly used a medical expert or compared statistical information of generated data vs real data to evaluate the quality of generated data. However, it is a very time consuming task to evaluate big databases by a medical expert and using only statistical information for comparing datasets is not sufficient since the structures/properties within a dataset cannot be fully captured. In this dissertation, WIE measure as a widely applicable measure across different datasets is developed and used to evaluate the quality of the generated patient data.

Probabilistic models are used to generate claims data attributes: Patterns and data structures of actual claims data are gathered and used in probabilistic models to generate claims data attributes. Probabilistic models are used to convert ELIX to ICD10, choosing primary diagnosis code, assigning HCPCS to claims, and generating R_NPI and P_NPI for claim table and line table. The desirable properties of probabilistic models are uncertainty quantification and structure exploitation of actual data. In addition, each time that IntPDG is applied a different set of patient claims data are generated due to probabilistic nature of IntPDG model.

6.2. Limitations and Future Works

To confirm the validity of generated data, couple of main steps are taken in this dissertation. First, hyperparameter optimization is performed and WIE measure evaluates the quality of generated patient level data. Next, to add other layers of data validation, summary statistics of generated claims data is compared with Medicare claims data to double confirm the consistency and transparency of the generated data. Although, other variables of claims including ICD10, HCPCS, specialty codes, and NPI are generated using probabilistic models which inherits structure exploitation of actual data, an evaluation method applicable on string variables can be applied to these variables as another layer of validation. These variables are string variables and WIE measure is only limited to the evaluation of binary data. In future works, a new evaluation method can be designed in a way that it will not be limited to evaluating binary attributes solely, but it can also be applicable to nominal and numeric variables.

Another limitation of this study is assigning NPI to patients' claims based on the state that both patient and provider reside. Although, this is a reasonable assumption that patients who seek procedures move within a state that they reside. However, the most precise approach that can be done in future work is to perform network analysis and find out patients' movement trend from actual data when they seek treatment.

For the purpose of this dissertation, a novel method, IntPDG, is developed for generating data and Medicare carrier claim tables are simulated in the presented work. In future work, IntPDG can be applied to generate other claim tables such as inpatient and outpatient claim tables. IntPDG can also be extended to generate other medical data such as EHR.

6.3. Conclusion

In this dissertation, a comprehensive study regarding health data generator approaches is presented and a novel method, IntPDG, is developed for generating patient claims data. Using IntPDG, three main tables from Medicare are simulated including:

Patient demographic table, carrier claim table and carrier line table. The developed data generator method can be used to generate any sizes and any types of claims data such as inpatient and outpatient claims data. Generated synthetic claims data is a rich source that includes information related to diagnoses, procedures, and utilization. While in some applications such as decision support systems it may not be possible, or advisable, to derive new knowledge directly from synthetic patient data, it can nevertheless be leveraged for a variety of secondary uses. Some applications of the generated claims data include: 1) Education: Since in many cases, it is not permitted to grant access real patient data to individuals, therefore, students can conduct numerous analyses using generated claims data for training and learning purposes. Generated claims data can be also used in training medical and nursing students. 2) Algorithm development: Algorithms and methods used in ML, data mining, health services research, statistics, and other areas need to be tested on data that closely resembles real patient data. While one cannot draw meaningful conclusions from results of applying these algorithms on generated claims data, they are often sufficient for the testing purposes and understanding limitations and properties of algorithms. 3) Software development and testing: Testing functionality of claim management system requires availability of patient claims data. Synthetic patient claims data are often sufficient for software development and performance testing. In fact, synthetic data allow for massive “stress testing” of software that may not be possible with real datasets with limited sizes. 4) Epidemiology and population health: Epidemiological models can be constructed using generated claims data to study how changes in a certain type of disease will affect a population. For example, changing the number of diabetes in generated claims data and we should see an increase in heart failure rate.

APPENDIX

Table 33 100 observations of the simulated demographic table

patid_gen	state	sex	race	dob	death_dt
10000000	10	1	1	10/24/1924	
10000001	45	1	1	11/14/1934	
10000002	45	1	1	2/26/1942	
10000003	5	2	4	12/12/1936	
10000004	38	2	1	6/20/1933	10/19/2011
10000005	14	1	1	10/4/1939	
10000006	52	1	2	1/16/1943	
10000007	33	2	5	6/14/1924	
10000008	14	1	4	3/29/1940	
10000009	6	2	5	2/9/1938	
10000010	49	1	2	2/10/1940	
10000011	24	1	1	6/1/1924	
10000012	37	1	1	6/15/1944	
10000013	22	1	1	12/29/1928	
10000014	14	2	1	1/24/1936	
10000015	38	2	4	2/12/1938	
10000016	24	2	1	5/11/1941	
10000017	17	1	1	3/21/1940	
10000018	3	2	1	12/30/1922	
10000019	39	1	1	9/19/1936	
10000020	45	1	1	1/11/1941	
10000021	5	1	2	2/22/1925	

10000022	33	2	1	7/29/1928	
10000023	5	2	1	8/19/1930	
10000024	5	2	1	9/23/1932	
10000025	24	1	5	10/5/1941	
10000026	33	2	5	8/16/1938	
10000027	5	1	5	12/14/1927	7/25/2010
10000028	10	2	6	6/7/1940	
10000029	44	2	1	5/14/1933	
10000030	52	2	5	8/6/1923	
10000031	37	1	1	9/5/1945	
10000032	25	1	1	10/5/1942	
10000033	45	2	1	4/9/1938	4/10/2010
10000034	3	2	1	5/6/1929	
10000035	5	1	1	3/20/1940	
10000036	31	1	1	12/27/1925	6/14/2011
10000037	19	1	2	8/22/1933	
10000038	5	2	1	7/30/1933	
10000039	23	2	1	5/13/1925	7/5/2011
10000040	6	2	5	9/29/1939	
10000041	31	1	5	9/10/1940	
10000042	31	2	1	4/25/1944	
10000043	41	1	1	4/21/1924	
10000044	5	1	1	11/27/1941	
10000045	52	2	5	7/23/1943	3/23/2010
10000046	24	2	1	9/16/1934	
10000047	26	1	1	02/29/1943	
10000048	45	1	5	7/15/1924	3/21/2010
10000049	5	1	3	2/3/1940	
10000050	43	2	1	4/1/1943	

10000051	11	2	1	11/16/1943	
10000052	11	1	1	12/30/1939	
10000053	22	2	5	6/3/1924	
10000054	39	2	1	11/13/1939	
10000055	33	2	1	8/11/1926	
10000056	52	1	5	6/6/1923	
10000057	43	1	1	10/19/1943	
10000058	33	1	1	1/16/1938	
10000059	5	2	2	4/10/1931	3/28/2011
10000060	38	1	1	7/24/1934	
10000061	38	2	5	1/27/1942	
10000062	49	1	1	12/22/1942	
10000063	11	1	1	3/26/1939	
10000064	5	1	1	1/22/1926	
10000065	14	1	2	8/2/1942	
10000066	45	1	1	3/27/1929	
10000067	46	1	1	4/23/1935	
10000068	32	2	1	12/1/1924	
10000069	36	2	1	3/25/1939	
10000070	14	1	1	6/26/1920	
10000071	14	2	1	6/12/1934	
10000072	23	2	1	3/26/1944	
10000073	49	1	1	5/30/1923	
10000074	19	2	2	11/24/1940	
10000075	5	1	1	8/7/1941	
10000076	23	1	1	2/22/1922	
10000077	33	1	2	4/20/1937	
10000078	11	1	2	3/8/1942	
10000079	3	1	1	10/3/1944	

10000080	44	1	1	6/3/1941	
10000081	14	2	2	6/13/1943	
10000082	10	1	1	4/14/1940	11/11/2011
10000083	49	2	1	2/12/1940	
10000084	50	2	1	12/21/1943	
10000085	5	1	1	4/14/1939	
10000086	45	2	5	1/29/1941	
10000087	29	2	1	3/21/1942	
10000088	23	2	1	4/14/1924	
10000089	32	1	1	8/11/1924	
10000090	42	1	2	9/19/1934	
10000091	39	2	2	6/10/1934	
10000092	45	1	1	2/17/1927	7/16/2011
10000093	15	2	1	9/17/1940	
10000094	6	2	5	1/30/1930	
10000095	4	1	1	7/4/1921	10/20/2011
10000096	14	2	5	10/22/1944	
10000097	36	1	1	10/12/1937	
10000098	6	2	1	12/4/1945	
10000099	22	2	5	8/29/1945	

Table 34 100 observations of generated claim table (empty diagnosis columns are dropped to fit the table into the document)

patid_gen	dob	sex	race	state	claimno	ref_npi	thru_dt	dgns_1	dgns_2	dgns_3	dgns_4	dgns_5
10000076	2/22/1922	1	1	23	1	500000013	1/1/2010	N183				
10000018	12/30/1922	2	1	3	2	100000000	1/1/2010	I10	Z992	J449	E890	

10000062	12/22/1942	1	1	49	3	100002709	1/1/2010	C61				
10000090	9/19/1934	1	2	42	4	100002233	1/2/2010	E119				
10000030	8/6/1923	2	5	52	5	500000034	1/2/2010	E039				
10000022	7/29/1928	2	1	33	6	500000020	1/2/2010	E119	D649			
10000096	10/22/1944	2	5	14	7	100000571	1/2/2010	E43				
10000056	6/6/1923	1	5	52	8	500000034	1/2/2010	E119				
10000002	2/26/1942	1	1	45	9	100002539	1/2/2010	C61	C787			
10000052	12/30/1939	1	1	11	10	500000007	1/3/2010	J449	G40301			
10000023	8/19/1930	2	1	5	11	500000003	1/3/2010	E039	D631			
10000021	2/22/1925	1	2	5	12	500000003	1/3/2010	C211				
10000041	9/10/1940	1	5	31	13	500000018	1/3/2010	E1165				
10000063	3/26/1939	1	1	11	14	500000007	1/3/2010	N186				
10000099	8/29/1945	2	5	22	15	100000932	1/4/2010	E039				
10000034	5/6/1929	2	1	3	16	100000002	1/4/2010	E119	F209			
10000007	6/14/1924	2	5	33	17	500000020	1/4/2010	F330	E039			
10000090	9/19/1934	1	2	42	18	500000026	1/5/2010	E119	F319			
10000023	8/19/1930	2	1	5	19	500000003	1/5/2010	E119	D649			
10000061	1/27/1942	2	5	38	20	500000023	1/5/2010	E039	E119			
10000034	5/6/1929	2	1	3	21	500000000	1/6/2010	F200				
10000031	9/5/1945	1	1	37	22	100001865	1/6/2010	F329				

10000042	4/25/1944	2	1	31	23	100001519	1/7/2010	G3184				
10000096	10/22/1944	2	5	14	24	100000561	1/7/2010	I10				
10000018	12/30/1922	2	1	3	25	100000014	1/7/2010	N189	I69959			
10000059	4/10/1931	2	2	5	26	500000003	1/7/2010	J449				
10000029	5/14/1933	2	1	44	27	500000028	1/8/2010	I10	E109			
10000032	10/5/1942	1	1	25	28	500000015	1/8/2010	E669				
10000030	8/6/1923	2	5	52	29	100002883	1/8/2010	E038	Z940			
10000057	10/19/1943	1	1	43	30	100002325	1/8/2010	E119				
10000063	3/26/1939	1	1	11	31	100000465	1/8/2010	N185				
10000007	6/14/1924	2	5	33	32	100001674	1/8/2010	F332	C8291			
10000007	6/14/1924	2	5	33	33	100001676	1/8/2010	E119	D47Z9	F329		
10000055	8/11/1926	2	1	33	34	500000020	1/8/2010	I5022				
10000027	12/14/1927	1	5	5	35	500000003	1/9/2010	E039	D649			
10000020	1/11/1941	1	1	45	36	500000030	1/9/2010	I10	N183			
10000031	9/5/1945	1	1	37	37	500000022	1/9/2010	E119				
10000022	7/29/1928	2	1	33	38	500000020	1/9/2010	E119	E6601			
10000012	6/15/1944	1	1	37	39	500000022	1/9/2010	N186	G309			
10000009	2/9/1938	2	5	6	40	100000324	1/10/2010	C7951				
10000014	1/24/1936	2	1	14	41	100000571	1/10/2010	I10				
10000084	12/21/1943	2	1	50	42	100002817	1/10/2010	R4182				

10000033	4/9/1938	2	1	45	43	500000030	1/10/2010	I714				
10000066	3/27/1929	1	1	45	44	100002524	1/11/2010	Z992				
10000080	6/3/1941	1	1	44	45	100002421	1/11/2010	N189				
10000078	3/8/1942	1	2	11	46	100000465	1/11/2010	I10				
10000030	8/6/1923	2	5	52	47	100002913	1/11/2010	N186	E018			
10000081	6/13/1943	2	2	14	48	100000560	1/11/2010	D631	I119			
10000075	8/7/1941	1	1	5	49	100000186	1/11/2010	C569				
10000001	11/14/1934	1	1	45	50	100002513	1/11/2010	I10				
10000058	1/16/1938	1	1	33	51	100001701	1/11/2010	I509	N184			
10000011	6/1/1924	1	1	24	52	100001118	1/12/2010	I10				
10000068	12/1/1924	2	1	32	53	100001581	1/12/2010	I10				
10000052	12/30/1939	1	1	11	54	500000007	1/12/2010	D649	M0579			
10000023	8/19/1930	2	1	5	55	500000003	1/12/2010	E039	D649			
10000058	1/16/1938	1	1	33	56	100001676	1/12/2010	I10	N183			
10000002	2/26/1942	1	1	45	57	100002511	1/13/2010	C3400				
10000064	1/22/1926	1	1	5	58	100000216	1/14/2010	Z992				
10000013	12/29/1928	1	1	22	59	500000012	1/14/2010	D649	I5023	F200	E039	E109
10000002	2/26/1942	1	1	45	60	100002511	1/14/2010	I129				
10000007	6/14/1924	2	5	33	61	500000020	1/14/2010	F329	E039			
10000044	11/27/1941	1	1	5	62	500000003	1/14/2010	F39				

10000099	8/29/1945	2	5	22	63	500000012	1/14/2010	E039	K279			
10000003	12/12/1936	2	4	5	64	100000188	1/15/2010	G40909				
10000071	6/12/1934	2	1	14	65	100000576	1/15/2010	G40009				
10000050	4/1/1943	2	1	43	66	500000027	1/15/2010	E119	E039			
10000051	11/16/1943	2	1	11	67	500000007	1/15/2010	I10	J449			
10000081	6/13/1943	2	2	14	68	100000560	1/16/2010	F10239				
10000081	6/13/1943	2	2	14	69	100000560	1/16/2010	I119				
10000077	4/20/1937	1	2	33	70	500000020	1/16/2010	I10				
10000045	7/23/1943	2	5	52	71	100002940	1/17/2010	E119	M0579	E039		
10000009	2/9/1938	2	5	6	72	500000005	1/17/2010	E6601				
10000065	8/2/1942	1	2	14	73	100000585	1/17/2010	Z992				
10000026	8/16/1938	2	5	33	74	500000020	1/17/2010	R569				
10000088	4/14/1924	2	1	23	75	100001050	1/18/2010	Z992				
10000060	7/24/1934	1	1	38	76	500000023	1/18/2010	I10				
10000092	2/17/1927	1	1	45	77	100002538	1/18/2010	N189				
10000030	8/6/1923	2	5	52	78	500000034	1/18/2010	I10	E039			
10000038	7/30/1933	2	1	5	79	500000003	1/18/2010	I10	E119			
10000023	8/19/1930	2	1	5	80	500000002	1/18/2010	E039	D509			
10000025	10/5/1941	1	5	24	81	500000014	1/18/2010	F325				
10000009	2/9/1938	2	5	6	82	500000005	1/18/2010	E119				

10000031	9/5/1945	1	1	37	83	500000022	1/19/2010	F331				
10000023	8/19/1930	2	1	5	84	100000197	1/19/2010	E119	I10			
10000064	1/22/1926	1	1	5	85	100000221	1/19/2010	E119				
10000089	8/11/1924	1	1	32	86	100001594	1/19/2010	E871				
10000042	4/25/1944	2	1	31	87	100001519	1/19/2010	N183				
10000074	11/24/1940	2	2	19	88	100000854	1/20/2010	I119	K7290			
10000052	12/30/1939	1	1	11	89	500000007	1/20/2010	G20				
10000025	10/5/1941	1	5	24	90	100001133	1/21/2010	F209	E669			
10000020	1/11/1941	1	1	45	91	100002541	1/21/2010	N186	I5023			
10000015	2/12/1938	2	4	38	92	500000023	1/21/2010	I739				
10000072	3/26/1944	2	1	23	93	500000013	1/21/2010	C800				
10000006	1/16/1943	1	2	52	94	500000034	1/21/2010	E119	J449			
10000030	8/6/1923	2	5	52	95	100002884	1/21/2010	I10	E039			
10000018	12/30/1922	2	1	3	96	500000000	1/21/2010	I10	E039	J449	Z992	
10000075	8/7/1941	1	1	5	97	500000003	1/22/2010	D538				
10000074	11/24/1940	2	2	19	98	500000011	1/22/2010	I1311				
10000090	9/19/1934	1	2	42	99	500000026	1/22/2010	E119	I509			
10000079	10/3/1944	1	1	3	100	100000002	1/22/2010	I10	G809			

Table 35 100 observations of generated line table

patid_gen	dob	sex	race	state	claimno	thru_dt	dgns_1	line_num	hcpcs	spclty	prf_npi
10000076	2/22/1922	1	1	23	1	1/1/2010	N183	1	80074	69	100001070
10000018	12/30/1922	2	1	3	2	1/1/2010	I10	1	3017F	11	100000000
10000018	12/30/1922	2	1	3	2	1/1/2010	I10	2	G8427	11	100000000
10000018	12/30/1922	2	1	3	2	1/1/2010	I10	3	99309	11	100000000
10000062	12/22/1942	1	1	49	3	1/1/2010	C61	3	81000	16	100002709
10000062	12/22/1942	1	1	49	3	1/1/2010	C61	2	99215	16	100002709
10000062	12/22/1942	1	1	49	3	1/1/2010	C61	1	84153	16	100002709
10000090	9/19/1934	1	2	42	4	1/2/2010	E119	1	83036	50	100002233
10000090	9/19/1934	1	2	42	4	1/2/2010	E119	2	84443	50	100002233
10000030	8/6/1923	2	5	52	5	1/2/2010	E039	1	84443	69	100002930
10000030	8/6/1923	2	5	52	5	1/2/2010	E039	4	99204	69	100002930
10000030	8/6/1923	2	5	52	5	1/2/2010	E039	2	84443	69	100002930
10000030	8/6/1923	2	5	52	5	1/2/2010	E039	3	84481	69	100002930
10000030	8/6/1923	2	5	52	5	1/2/2010	E039	5	99213	69	100002930
10000022	7/29/1928	2	1	33	6	1/2/2010	E119	5	36410	69	100001721
10000022	7/29/1928	2	1	33	6	1/2/2010	E119	4	2027F	69	100001721
10000022	7/29/1928	2	1	33	6	1/2/2010	E119	3	85025	69	100001721
10000022	7/29/1928	2	1	33	6	1/2/2010	E119	2	80053	69	100001721
10000022	7/29/1928	2	1	33	6	1/2/2010	E119	1	P9603	69	100001721
10000096	10/22/1944	2	5	14	7	1/2/2010	E43	1	99232	6	100000571
10000056	6/6/1923	1	5	52	8	1/2/2010	E119	1	99490	6	100002896
10000056	6/6/1923	1	5	52	8	1/2/2010	E119	2	82962	6	100002896
10000002	2/26/1942	1	1	45	9	1/2/2010	C61	2	J9217	34	100002539
10000002	2/26/1942	1	1	45	9	1/2/2010	C61	1	99231	34	100002539
10000052	12/30/1939	1	1	11	10	1/3/2010	J449	1	99214	8	100000467
10000052	12/30/1939	1	1	11	10	1/3/2010	J449	2	99214	8	100000467
10000052	12/30/1939	1	1	11	10	1/3/2010	J449	3	99215	8	100000467
10000023	8/19/1930	2	1	5	11	1/3/2010	E039	1	85025	69	100000233
10000021	2/22/1925	1	2	5	12	1/3/2010	C211	3	84100	92	100000227

10000021	2/22/1925	1	2	5	12	1/3/2010	C211	2	77014	92	100000227
10000021	2/22/1925	1	2	5	12	1/3/2010	C211	1	77301	92	100000227
10000041	9/10/1940	1	5	31	13	1/3/2010	E1165	1	99213	69	100001535
10000041	9/10/1940	1	5	31	13	1/3/2010	E1165	2	84443	69	100001535
10000063	3/26/1939	1	1	11	14	1/3/2010	N186	1	G9500	30	100000474
10000099	8/29/1945	2	5	22	15	1/4/2010	E039	1	36415	8	100000932
10000034	5/6/1929	2	1	3	16	1/4/2010	E119	1	36415	8	100000002
10000007	6/14/1924	2	5	33	17	1/4/2010	F330	2	82607	80	100001691
10000007	6/14/1924	2	5	33	17	1/4/2010	F330	1	90834	80	100001691
10000090	9/19/1934	1	2	42	18	1/5/2010	E119	1	92014	69	100002279
10000090	9/19/1934	1	2	42	18	1/5/2010	E119	2	81001	69	100002279
10000023	8/19/1930	2	1	5	19	1/5/2010	E119	1	80061	90	100000231
10000023	8/19/1930	2	1	5	19	1/5/2010	E119	2	85027	90	100000231
10000061	1/27/1942	2	5	38	20	1/5/2010	E039	1	86235	69	100002000
10000061	1/27/1942	2	5	38	20	1/5/2010	E039	3	84443	69	100002000
10000061	1/27/1942	2	5	38	20	1/5/2010	E039	2	36415	69	100002000
10000034	5/6/1929	2	1	3	21	1/6/2010	F200	1	99232	69	100000047
10000034	5/6/1929	2	1	3	21	1/6/2010	F200	2	1036F	69	100000047
10000034	5/6/1929	2	1	3	21	1/6/2010	F200	3	P9603	69	100000047
10000034	5/6/1929	2	1	3	21	1/6/2010	F200	4	99233	69	100000047
10000034	5/6/1929	2	1	3	21	1/6/2010	F200	5	99231	69	100000047
10000034	5/6/1929	2	1	3	21	1/6/2010	F200	6	90834	69	100000047
10000031	9/5/1945	1	1	37	22	1/6/2010	F329	1	G8784	93	100001865
10000042	4/25/1944	2	1	31	23	1/7/2010	G3184	1	99212	83	100001519
10000096	10/22/1944	2	5	14	24	1/7/2010	I10	1	G8427	97	100000561
10000018	12/30/1922	2	1	3	25	1/7/2010	N189	1	36415	26	100000014
10000018	12/30/1922	2	1	3	25	1/7/2010	N189	2	99334	26	100000014
10000018	12/30/1922	2	1	3	25	1/7/2010	N189	3	83550	26	100000014
10000018	12/30/1922	2	1	3	25	1/7/2010	N189	4	A0428	26	100000014
10000059	4/10/1931	2	2	5	26	1/7/2010	J449	1	99214	34	100000214
10000029	5/14/1933	2	1	44	27	1/8/2010	I10	4	G8417	69	100002465

10000029	5/14/1933	2	1	44	27	1/8/2010	I10	1	93010	69	100002465
10000029	5/14/1933	2	1	44	27	1/8/2010	I10	2	G0403	69	100002465
10000029	5/14/1933	2	1	44	27	1/8/2010	I10	3	P9603	69	100002465
10000032	10/5/1942	1	1	25	28	1/8/2010	E669	1	G0479	9	100001267
10000030	8/6/1923	2	5	52	29	1/8/2010	E038	1	99214	11	100002883
10000030	8/6/1923	2	5	52	29	1/8/2010	E038	3	84480	11	100002883
10000030	8/6/1923	2	5	52	29	1/8/2010	E038	2	84439	11	100002883
10000057	10/19/1943	1	1	43	30	1/8/2010	E119	2	83721	11	100002325
10000057	10/19/1943	1	1	43	30	1/8/2010	E119	1	80048	11	100002325
10000063	3/26/1939	1	1	11	31	1/8/2010	N185	2	4255F	11	100000465
10000063	3/26/1939	1	1	11	31	1/8/2010	N185	1	36415	11	100000465
10000063	3/26/1939	1	1	11	31	1/8/2010	N185	3	99223	11	100000465
10000007	6/14/1924	2	5	33	32	1/8/2010	F332	1	G8431	11	100001674
10000007	6/14/1924	2	5	33	32	1/8/2010	F332	2	99232	11	100001674
10000007	6/14/1924	2	5	33	33	1/8/2010	E119	4	99214	8	100001676
10000007	6/14/1924	2	5	33	33	1/8/2010	E119	1	82948	8	100001676
10000007	6/14/1924	2	5	33	33	1/8/2010	E119	5	83036	8	100001676
10000007	6/14/1924	2	5	33	33	1/8/2010	E119	2	99215	8	100001676
10000007	6/14/1924	2	5	33	33	1/8/2010	E119	3	82248	8	100001676
10000055	8/11/1926	2	1	33	34	1/8/2010	I5022	1	93000	29	100001700
10000055	8/11/1926	2	1	33	34	1/8/2010	I5022	4	1036F	29	100001700
10000055	8/11/1926	2	1	33	34	1/8/2010	I5022	2	99232	29	100001700
10000055	8/11/1926	2	1	33	34	1/8/2010	I5022	3	99231	29	100001700
10000027	12/14/1927	1	5	5	35	1/9/2010	E039	3	36415	69	100000233
10000027	12/14/1927	1	5	5	35	1/9/2010	E039	4	84439	69	100000233
10000027	12/14/1927	1	5	5	35	1/9/2010	E039	1	P9603	69	100000233
10000027	12/14/1927	1	5	5	35	1/9/2010	E039	2	84480	69	100000233
10000020	1/11/1941	1	1	45	36	1/9/2010	I10	3	80061	69	100002558
10000020	1/11/1941	1	1	45	36	1/9/2010	I10	1	84439	69	100002558
10000020	1/11/1941	1	1	45	36	1/9/2010	I10	2	80061	69	100002558
10000020	1/11/1941	1	1	45	36	1/9/2010	I10	5	80053	69	100002558

10000020	1/11/1941	1	1	45	36	1/9/2010	I10	4	G8752	69	100002558
10000031	9/5/1945	1	1	37	37	1/9/2010	E119	2	99214	69	100001907
10000031	9/5/1945	1	1	37	37	1/9/2010	E119	3	80061	69	100001907
10000031	9/5/1945	1	1	37	37	1/9/2010	E119	4	82043	69	100001907
10000031	9/5/1945	1	1	37	37	1/9/2010	E119	1	83036	69	100001907
10000022	7/29/1928	2	1	33	38	1/9/2010	E119	1	84311	69	100001721
10000022	7/29/1928	2	1	33	38	1/9/2010	E119	2	36415	69	100001721
10000012	6/15/1944	1	1	37	39	1/9/2010	N186	1	99232	10	100001882
10000009	2/9/1938	2	5	6	40	1/10/2010	C7951	3	J0897	90	100000324

Table 36 Mean and standard deviation of count lines for each primary diagnosis; obtained from actual line table (50 observations)

Primary diagnosis	Average line number	Standard deviation
L03049	1.476923077	0.786574
T82828S	1.684210526	0.948829
T424X6A	2	0.816497
M41116	2.125	2.619041
H26493	1.640585541	1.161028
E113219	1.876190476	1.671585
S53429A	1.25	0.433013
S42035S	1	0
T24391A	1.684210526	1.126365
S59902D	1.954545455	2.163273
C9211	1.94625323	1.70974
M6751	1.390243902	1.112634
S79119A	1.333333333	0.471405
M4125	2.232053422	2.063979
S4992XS	1.633333333	1.353596
S62024S	2	1.224745

M205X9	1.324455206	0.959877
S93325S	1.272727273	0.445362
M05572	1.65625	1.134939
Z842	2.535714286	2.306546
M832	1.914285714	2.061503
N3090	1.811710806	1.43569
M898X1	1.93270366	1.837933
S52011A	1.333333333	0.745356
S66302D	1	0
T24202D	1.565217391	1.555526
M71311	1.692307692	0.866879
S93522D	1	0
M80832D	1.25	0.433013
I4460	1.340707965	1.145915
S06340S	2.928571429	2.086155
S72131A	1.7	1.16619
S92513G	1	0
O2230	1.090909091	0.28748
Z8774	1.666666667	1.30227
S82875A	1.225806452	0.418112
F642	1.052631579	0.223297
S72042S	1.230769231	0.478327
S63131A	2	0.707107
B852	1.228571429	0.795908
T25299A	2	0
H5021	1.694267516	1.143592
G935	1.446504993	1.000174
I160	1.438041815	0.965156
S65502A	1	0
T413X1A	1.090909091	0.28748
S92341S	1.714285714	1.385051

M19231	2	1.839288
Z170	1.77262181	1.507689
C4082	2.333333333	0.942809

Table 37 Probability that each primary diagnosis can show up with HCPCS codes; obtained from actual line table (50 observations)

dgns_1	hcpcs_cd	probability
A047	82725	2.28E-05
A047	G8753	9.11E-05
A047	70450	6.83E-05
A047	80500	2.28E-05
A047	82715	6.83E-05
A047	85045	2.28E-05
A047	99284	0.004985317
A047	4025F	2.28E-05
A047	R0070	2.28E-05
A047	87272	0.00011382
A047	A0434	4.55E-05
A047	76942	2.28E-05
A047	A6259	0.00022764
A047	A0427	9.11E-05
A047	87102	4.55E-05
A047	P9603	0.002686153

A047	85048	4.55E-05
A047	47600	2.28E-05
A047	97802	2.28E-05
A047	80061	9.11E-05
A047	G8752	0.000523572
A047	87324	0.013635639
A047	G0471	0.002572333
A047	4256F	2.28E-05
A047	99232	0.277311116
A047	78807	2.28E-05
A047	G0408	2.28E-05
A047	J1100	4.55E-05
A047	G9655	4.55E-05
A047	36416	0.000136584
A047	44150	0.000728448
A047	1036F	0.004712149
A047	99053	0.00011382
A047	A4222	0.00022764
A047	G8599	9.11E-05
A047	74270	4.55E-05
A047	J3490	2.28E-05

A047	G0444	4.55E-05
A047	G8598	0.00034146
A047	76705	2.28E-05
A047	80162	4.55E-05
A047	G8600	6.83E-05
A047	G8734	4.55E-05
A047	82310	6.83E-05
A047	93460	4.55E-05
A047	G0427	6.83E-05
A047	84075	2.28E-05
A047	45378	0.002481277
A047	82150	4.55E-05
A047	88108	9.11E-05

Table 38 Specialty codes in Medicare data

Specialty Code	Description
01	General practice
02	General surgery
03	Allergy/immunology
04	Otolaryngology
05	Anesthesiology
06	Cardiology
07	Dermatology

08	Family practice
09	Interventional pain management
10	Gastroenterology
11	Internal medicine
12	Osteopathic manipulative medicine
13	Neurology
14	Neurosurgery
15	Speech language pathologist in private practice
16	Obstetrics/gynecology
17	Hospice and palliative care
18	Ophthalmology
19	Oral surgery (dentists only)
20	Orthopedic surgery
21	Cardiac electrophysiology
22	Pathology
23	Sports medicine
24	Plastic and reconstructive surgery
25	Physical medicine and rehabilitation
26	Psychiatry
27	Geriatric psychiatry
28	Colorectal surgery (formerly proctology)
29	Pulmonary disease
30	Diagnostic radiology
31	Intensive cardiac rehabilitation
32	Anesthesiologist assistant
33	Thoracic surgery
34	Urology
35	Chiropractic

36	Nuclear medicine
37	Pediatric medicine
38	Geriatric medicine
39	Nephrology
40	Hand surgery
41	Optometry
42	Certified nurse midwife
43	Certified registered nurse anesthetist (CRNA)
44	Infectious disease
45	Mammography screening center
46	Endocrinology
47	Independent diagnostic testing facility
48	Podiatry
49	Ambulatory surgical center
50	Nurse practitioner
51	Medical supply company with certified orthotist
52	Medical supply company with certified prosthetist
53	Medical supply company with certified prosthetist-orthotist
54	Medical supply company not included in specialties 51-53
55	Individual orthotic personnel certified by an accrediting organization
56	Individual prosthetic personnel certified by an accrediting organization
57	Individual prosthetic/orthotic personnel certified by an accrediting organization
58	Medical supply company with registered pharmacist
59	Ambulance service (private)
60	Public health or welfare agencies (federal, state, and local)
61	Voluntary health or charitable agencies (e.g., National Cancer Society, National Heart Association, Catholic Charities)
62	Psychologist (billing independently)

63	Portable x-ray supplier (billing independently)
64	Audiologist (billing independently)
65	Physical therapist in private practice
66	Rheumatology
67	Occupational therapist in private practice
68	Clinical psychologist
69	Clinical laboratory (billing independently)
70	Single or Multi-specialty clinic or group practice (PA Group)
71	Registered dietician/nutrition professional
72	Pain management
73	Mass immunization roster biller
74	Radiation therapy center
76	Peripheral vascular disease
77	Vascular surgery
78	Cardiac surgery
79	Addiction medicine
80	Licensed clinical social worker
81	Critical care (intensivists)
82	Hematology
83	Hematology/oncology
84	Preventation medicine
85	Maxillofacial surgery
86	Neuropsychiatry
87	All other suppliers, e.g., drug stores
88	Unknown provider
89	Certified clinical nurse specialist
90	Medical oncology
91	Surgical oncology

92	Radiation oncology
93	Emergency medicine
94	Interventional radiology
95	Unknown supplier
96	Optician
97	Physician assistant
98	Gynecological/oncology
99	Unknown physician specialty
A0	Hospital
A1	Skilled nursing facility
A2	Intermediate care nursing facility
A3	Nursing facility, other
A4	Home health agency
A5	Pharmacy
A6	Medical supply company with respiratory therapist
A7	Department store
A8	Grocery store
B1	Oxygen/Oxygen Related Equipment
B2	Pedorthic personnel
B3	Medical supply company with pedorthic personnel
B4	Rehabilitation agency
B5	Ocularist
C0	Sleep medicine
C1	Centralized flu
C2	Indirect payment procedure
C3	Interventional cardiology
C4	Restricted use
C5	Dentist

C6	Hospitalist
C7	Advanced heart failure and transplant cardiology
C8	Medical toxicology
C9	Hematopoietic cell transplantation and cellular therapy
D1	Medicare Diabetes Prevention Program
D3	Medical genetics and genomics
D4	Undersea and Hyperbaric Medicine
D5	Opioid Treatment Program

Table 39 Ratio of service providers' population for each specialty form actual claim line table

Specialty code	Ratio of P-NPI for each specialty
11	0.098243177
50	0.091065559
8	0.080329577
97	0.06107826
65	0.04163856
93	0.041164746
43	0.039333235
5	0.036093602
35	0.031774091
30	0.028843675
73	0.027850377
41	0.027419378
16	0.024593619
6	0.021110419
26	0.020807863
20	0.020415872

2	0.019170445
80	0.017620559
18	0.016876537
68	0.014374266
48	0.014344771
13	0.013342911
10	0.012694033
22	0.011117507
7	0.010941492
C1	0.009893011
29	0.009603775
59	0.008841677
34	0.008588595
4	0.008450637
39	0.008191847
83	0.008026298
25	0.007438311
64	0.005962637
1	0.005541152
46	0.005331836
44	0.005304245
49	0.004924623
14	0.004371839
66	0.004339491
67	0.004260522
92	0.004215804
24	0.00402076
81	0.003183498

3	0.003159712
90	0.003077889
77	0.003046492
69	0.00280673
C3	0.002705878
37	0.002697316
47	0.002626909
71	0.002208278
72	0.002178784
70	0.002154047
33	0.002102669
89	0.00194378
21	0.001922848
38	0.001878131
9	0.001723998
32	0.001707824
94	0.001627904
78	0.001449985
40	0.001385288
28	0.001379579
19	0.001354842
42	0.001088441
23	0.000990443
17	0.000944774
98	0.00092194
15	0.000898154
91	0.000867708
85	0.000824893

87	0.000775419
82	0.000706915
12	0.000634607
36	0.000585132
99	0.000450028
C0	0.000360594
84	0.000340613
63	0.000311119
62	0.000306362
60	0.000286382
27	0.000210267
79	0.000162695
C6	0.000144618
86	0.000127492
76	7.14E-05
74	4.76E-05
45	2.57E-05
75	2.19E-05
88	1.62E-05
C5	3.81E-06
58	9.51E-07

Table 40 Specialty and rate of claims having different NPI for each specialty; obtained from actual claim tables

Specialty code	Percent claims with different NPI
58	1
75	1
69	0.999807128

63	0.999547279
47	0.995110374
74	0.994040194
65	0.983344255
30	0.980257653
67	0.968683078
22	0.957926395
45	0.948154657
32	0.923893541
43	0.909564697
15	0.909190639
94	0.907206235
64	0.90590634
49	0.899545177
71	0.884403546
36	0.881724895
C5	0.8
05	0.799254292
92	0.749000896
88	0.684210526
10	0.608040201
77	0.601268159
76	0.593047902
06	0.549359158
13	0.540859191
44	0.539433182
21	0.533377173
14	0.532156443

78	0.531817195
C3	0.53034327
85	0.526409145
33	0.515702109
28	0.507289842
C0	0.504217119
19	0.503891277
09	0.502688208
99	0.499201146
48	0.48373686
29	0.482963
04	0.470360532
87	0.461077309
02	0.445515068
91	0.441277473
72	0.438243566
82	0.429978947
34	0.427040227
03	0.426068259
39	0.425227518
24	0.413649975
25	0.408901423
98	0.403134796
97	0.401680234
81	0.390878185
66	0.384700248
90	0.384491231
83	0.382228867

17	0.367710508
40	0.364973819
46	0.364918415
50	0.330638096
20	0.330393912
86	0.319520881
23	0.314178127
42	0.292326067
89	0.291245061
07	0.27450478
18	0.260736682
84	0.252952144
68	0.238197417
37	0.236125991
16	0.213188648
27	0.190882833
12	0.170888409
C6	0.151376147
80	0.134970605
41	0.134599688
11	0.132116374
26	0.128504745
62	0.127705164
79	0.1245571
60	0.105545617
38	0.092885298
01	0.089149979
08	0.071710714

93	0.051581488
59	0.034475614
35	0.029878388
73	0.016354969
70	0.01538729
C1	1.25E-05

REFERENCES

Abouelmehdi, K., Beni-Hessane, A., & Khaloufi, H. (2018). Big healthcare data: preserving security and privacy. Journal of Big Data, 5(1), 1.

Alzheimer's Prevention Registry. (2020). Available online at: https://www.endalznow.org/

Ancker, J. S., Kern, L. M., Edwards, A., Nosal, S., Stein, D. M., Hauser, D., ... & with the HITEC Investigators. (2014). How is the electronic health record being used? Use of EHR data to assess physician-level variability in technology use. Journal of the American Medical Informatics Association, 21(6), 1001-1008.

Baowaly, M. K., Lin, C. C., Liu, C. L., & Chen, K. T. (2018). Synthesizing electronic health records using improved generative adversarial networks. Journal of the American Medical Informatics Association, 26(3), 228-241.

Blackwell, D. L., Lucas, J. W., & Clarke, T. C. (2014). Summary health statistics for US adults: national health interview survey, 2012. Vital and health statistics. Series 10, Data from the National Health Survey, (260), 1-161.

Buczak, A. L., Babin, S., & Moniz, L. (2010). Data-driven approach for creating synthetic electronic medical records. BMC Medical Informatics and Decision Making, 10(1), 59.

Camino, R., Hammerschmidt, C., & State, R. (2018). Generating multi-categorical samples with generative adversarial networks. arXiv preprint arXiv:1807.01202. Camino et al., 2018

Carlitz, L. (1948). $ q $-Bernoulli numbers and polynomials. Duke Mathematical Journal, 15(4), 987-1000.

CDC. (2015). International classification of diseases, (ICD-10-CM/PCS) transition - background

CDC. (2019). Behavioral risk factor surveillance system. Available online at: https://www.cdc.gov/brfss/

Census. (2020). Demographic data. Available online at: https://www.census.gov/programs-surveys/ces/data/restricted-use-data/demographic-data.html

Children’s Health Foundation. (2013). Available online at: http://www.ch-foundation.org/improving-pediatric-practices/quality-improvement/pediatric-asthma-care-management/pediatric-asthma-registry

Choi, E., Biswal, S., Malin, B., Duke, J., Stewart, W. F., & Sun, J. (2017). Generating multi-label discrete patient records using generative adversarial networks. arXiv preprint arXiv:1703.06490.

CMS. (2014). Electronic health records. Available online at: https://www.cms.gov/Medicare/E-Health/EHealthRecords

CMS. (2019). Standard analytical files (Medicare claims) – LDS. Available online at: https://www.cms.gov/Research-Statistics-Data-and-Systems/Files-for-Order/LimitedDataSets/StandardAnalyticalFiles

CMS. (2020). Medicare claims synthetic public use files. Available online at: https://www.cms.gov/Research-Statistics-Data-and-Systems/Downloadable-Public-Use-Files/SynPUFs CMS. (2020)

Dash, S., Dutta, R., Guyon, I., Pavao, A., Yale, A., & Bennett, K. P. (2019). Synthetic event time series health data generation. arXiv preprint arXiv:1911.06411. Dash et al., 2019

Del Carmen Rodríguez-Hernández, M., Ilarri, S., Hermoso, R., & Trillo-Lado, R. (2017). DataGenCARS: A generator of synthetic data for the evaluation of context-aware recommendation systems. Pervasive and Mobile Computing, 38, 516-541.

DHS. (2020). Demographic and health survey. Available online at: https://dhsprogram.com/what-we-do/survey-Types/dHs.cfm

Dube, K., & Gallagher, T. (2013, August). Approach and method for generating realistic synthetic electronic healthcare records for secondary use. In International Symposium on Foundations of Health Informatics Engineering and Systems (pp. 69-86). Springer, Berlin, Heidelberg.

Elixhauser, A., Steiner, C., Harris, D. R., & Coffey, R. M. (1998). Comorbidity measures for use with administrative data. Medical care, 8-27.

Esposito, C., De Santis, A., Tortora, G., Chang, H., & Choo, K. K. R. (2018). Blockchain: A panacea for healthcare cloud-based data security and privacy?. IEEE Cloud Computing, 5(1), 31-37.

Gagne, J. J., Glynn, R. J., Avorn, J., Levin, R., & Schneeweiss, S. (2011). A combined comorbidity score predicted mortality in elderly patients better than existing scores. Journal of clinical epidemiology, 64(7), 749-759.

Gal, Y., Chen, Y., & Ghahramani, Z. (2015). Latent Gaussian processes for distribution estimation of multivariate categorical data.   Gal et al., 2015

Goncalves, A., Ray, P., Soper, B., Stevens, J., Coyle, L., & Sales, A. P. (2020). Generation and evaluation of synthetic patient data. BMC Medical Research Methodology, 20, 1-40.   Goncalves et al., 2020

Guibas, J. T., Virdi, T. S., & Li, P. S. (2017). Synthetic medical images from dual generative adversarial networks. arXiv preprint arXiv:1709.01872.

HCUP. (2019). Clinical classifications software for ICD-10-PCS. Available online at: https://www.hcup-us.ahrq.gov/toolssoftware/ccs10/ccs10.jsp

He, D., Mathews, S. C., Kalloo, A. N., & Hutfless, S. (2014). Mining high-dimensional administrative claims data to predict early hospital readmissions. Journal of the American Medical Informatics Association, 21(2), 272-279.

Hu, J., Dong, Y., Chen, X., Liu, Y., Ma, D., Liu, X., ... & He, W. (2015). Prevalence of suicide attempts among Chinese adolescents: a meta-analysis of cross-sectional studies. Comprehensive Psychiatry, 61, 78-89.

Ippolito, P. P. (2019). Hyperparameters optimization. Available online at: https://towardsdatascience.com/hyperparameters-optimization-526348bb8e2d
Ippolito, 2019

Jensen, P. B., Jensen, L. J., & Brunak, S. (2012). Mining electronic health records: towards better research applications and clinical care. Nature Reviews Genetics, 13(6), 395.

Kavak, H., Kim, J. S., Crooks, A., Pfoser, D., Wenk, C., & Züfle, A. (2019, August). Location-based social simulation. In Proceedings of the 16th International Symposium on Spatial and Temporal Databases (pp. 218-221). ACM.

Kofinas, D. T., Spyropoulou, A., & Laspidou, C. S. (2018). A methodology for synthetic household water consumption data generation. Environmental modelling & software, 100, 48-66.

Kostkova, P., Brewer, H., de Lusignan, S., Fottrell, E., Goldacre, B., Hart, G., ... & Ross, E. (2016). Who owns the data? Open data for healthcare. Frontiers in public health, 4, 7.

Maciejewski, R., Hafen, R., Rudolph, S., Tebbetts, G., Cleveland, W. S., Grannis, S. J., & Ebert, D. S. (2009). Generating synthetic syndromic-surveillance data for evaluating visual-analytics techniques. IEEE Computer Graphics and Applications, 29(3), 18-28.

NIH. (2019). Brief Description of the MHOS Surveys. Available online at: https://healthcaredelivery.cancer.gov/seer-mhos/instruments/

NIH. (2020a). List of registries. Available online at: https://www.nih.gov/health-information/nih-clinical-research-trials-you/list-registries

NIH. (2020b). SEER-Medicare: Medicare claims files. Available online at: https://healthcaredelivery.cancer.gov/seermedicare/medicare/claims.html

NIH-UMLS. (2019). Unified medical language system. Available online at: https://www.nlm.nih.gov/research/umls/index.html

Obermeyer, Z., Cohn, B., Wilson, M., Jena, A. B., & Cutler, D. M. (2017). Early death after discharge from emergency departments: analysis of national US insurance claims data. bmj, 356, j239.

Paul, S. (2018). Hyperparameter optimization in machine learning models. Available online at: https://www.datacamp.com/community/tutorials/parameter-optimization-machine-learning-models

Pereira, J. (2020). HCPCS codes. Available online at: https://www.medicalbillingandcoding.org/hcpcs-codes/

Quan, H., Sundararajan, V., Halfon, P., Fong, A., Burnand, B., Luthi, J. C., ... & Ghali, W. A. (2005). Coding algorithms for defining comorbidities in ICD-9-CM and ICD-10 administrative data. Medical care, 1130-1139.

Schauer, M. (2015). International encyclopedia of social & behavioral sciences. Narrative exposure therapy, 2nd edn. Elsevier, Amsterdam.

SEER. (2020). About the SEER registries. Available online at: https://seer.cancer.gov/registries/

Singh, A., Nadkarni, G., Gottesman, O., Ellis, S. B., Bottinger, E. P., & Guttag, J. V. (2015). Incorporating temporal EHR data in predictive models for risk stratification of renal function deterioration. Journal of biomedical informatics, 53, 220-228.

Srivastava, T. (2015). Tuning the parameters of your random forest model. Analytics Vidhya, 9. Srivastava, 2015

Stein, J. D., Lum, F., Lee, P. P., Rich III, W. L., & Coleman, A. L. (2014). Use of health care claims data to study patients with ophthalmologic conditions. Ophthalmology, 121(5), 1134-1141.

Titchmarsh, E. C. T., Titchmarsh, E. C., & Heath-Brown, D. R. (1986). The theory of the Riemann zeta-function. Oxford University Press.

Trustees Report & Trust Funds (2019). Available online at: https://www.cms.gov/Research-Statistics-Data-and-Systems/Statistics-Trends-and-Reports/ReportsTrustFunds

Tyree, P. T., Lind, B. K., & Lafferty, W. E. (2006). Challenges of using medical insurance claims data for utilization analysis. American Journal of Medical Quality, 21(4), 269-275.

Varese, F., Smeets, F., Drukker, M., Lieverse, R., Lataster, T., Viechtbauer, W., ... & Bentall, R. P. (2012). Childhood adversities increase the risk of psychosis: a meta-analysis of patient-control, prospective-and cross-sectional cohort studies. Schizophrenia bulletin, 38(4), 661-671.

Vestal, C. (2014). Can Claims Data Crack the Health Care Cost Riddle?. USA Today.

Walonoski, J., Kramer, M., Nichols, J., Quina, A., Moesel, C., Hall, D., ... & McLachlan, S. (2017). Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. Journal of the American Medical Informatics Association, 25(3), 230-238.

Wilson, J., & Bock, A. (2012). The benefit of using both claims data and electronic medical record data in health care analysis. Optum Insight, 1-4.

Wojtusiak, J. (2016). Towards intelligent patient data generator, Reports of the Machine Learning and Inference Laboratory, MLI 16-2.

Zare, M., & Wojtusiak, J. (2018, December). Weighted Itemsets Error (WIE) approach for evaluating generated synthetic patient data. In 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 1017-1022). IEEE.

BIOGRAPHY

Mojtaba Zare graduated from Amini High School, Marvdasht, Iran in 2007. He received his Bachelor of Electrical Engineering from Babol Noshirvani University of Technology in 2011, and received his master in Information Technology from University Technology Malaysia in 2015. He was employed as a Graduate Research Assistant at George Mason University since the start of his Ph.D. and received his Ph.D. in Health Services Research with Knowledge Discovery and Health Informatics Concentration from George Mason University in 2020.