ANALYSES OF MEDICAL DEVICE FAILURES RELATED TO COMPUTING TECHNOLOGY

By

Deepak Khanal

A Dissertation Submitted

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy in Biomedical Informatics

School of Health Professions

Biomedical and Health Sciences

Rutgers, the State University of New Jersey

May 2018

APPROVALS

ANALYSES OF MEDICAL DEVICE FAILURES RELATED TO COMPUTING TECHNOLOGY

By

Deepak Khanal

Dissertation Committee:

Shankar Srinivasan, PhD, Rutgers University

Mohamed F. AbdelHady, PhD, Microsoft

Suril Gohel, PhD, Rutgers University

Edwin O. Heierman III, PhD, Abbott

Approved by the Dissertation Committee:

_____ Date: _____

_____ Date: _____

_____ Date: _____

_____ Date: _____

ABSTRACT

ANALYSES OF MEDICAL DEVICE FAILURES RELATED TO COMPUTING TECHNOLOGY

By

Deepak Khanal

Background: The adoption of computing technology in modern medical devices is ubiquitous. However, limited research currently exists on the role of computing technology on medical device failures and patient safety. The U.S. Food and Drug Administration (FDA) collects and publishes reports of medical device events as a part of its Medical Device Reporting program, but the problem codes assigned to events in the published database do not appear to be reliable to identify computing technology-related events.

Methods: A supervised machine learning technique was designed and implemented to classify over 11 million natural-language narratives of medical device events reported to the FDA between 2007 and 2016 to identify events related to computing technology. The result of the classification was then used to analyze the events from several dimensions.

Results: A total of 5,110,200 reports of medical device events were submitted to the FDA between 2007 and 2016. Of these, 1,155,516 (22.61%) were related to computing technology. Number of computing technology-related medical device events reported to the FDA jumped nearly 7-fold from 37,679 in 2007 to 262,407 in 2016. Nearly all (99.36%) of these reports were submitted to the FDA by the manufacturers of devices, even though patients were the original reporters of the issues leading up to the submission in nearly a third (32.46%) of the events. A total of 3,449 medical device events related to computing

technology were associated with patient deaths in the 10-year period. Also, events published by the FDA on its Manufacturer and User Facility Device Experience (MAUDE) database were found to be missing problem codes in 62.74% of a sampled set (N=102) of events related to computing technology and inaccurate in 26.47% of the sampled events.

Conclusions: Computing technology-related events constitute a significant portion of medical device events reported to the FDA every year. Overall, these events are on an increasing trend on an absolute basis. Manufacturers are the submitters of nearly all of the computing technology-related medical device events reported to the FDA. Medical device events related to computing technology can cause serious adverse patient events, including death. Problem codes assigned to computing technology-related medical device events in the MAUDE database published by the FDA are inaccurate at a significant rate and should not be used in research.

ACKNOWLEDGEMENTS

I am deeply grateful for all the support I have received over the years since I embarked on this journey. I am particularly fortunate to have Dr. Shankar Srinivasan as my advisor, who challenged, guided and supported me throughout this project. I am also thankful to my dissertation committee composed of accomplished scholars from academia and the industry. Thank you Dr. Mohamed F. AbdelHady for providing me with the guidance throughout this project as I explored the worlds of artificial intelligence and machine learning. Dr. Edwin O. Heierman III, thank you for the guidance and very concrete help in staying focused on the practical values of this research. Thanks also to Dr. Suril Gohel for his important feedback and advice in the course of this project.

Special thanks to my family for enabling me to work on this project. I would, especially, like to acknowledge my parents Mr. Hari P. and Mrs. Sita D. Khanal, who not only instilled my passion for learning, but also came from far to help with some of the household responsibilities that I could not attend to due to my studies. Thanks also to my wife and my daughters for supporting me on this project in ways I cannot express in words.

I would also like to thank Dr. Neil W. Rickert, a brilliant computer scientist and mathematician, and one of the best professors I have had in all of my academic life for accepting my request to review this work, and for the feedback. I would also like to express my gratitude to my colleagues, who supported my aspirations for continuous education. In particular, I would like to thank my supervisors, past and present: Ms. Carol Anderson, Mr. Adam F. Levar, Mr. David M. Peters, Mr. Rodney J. Rasmussen, Mr. Jayashankar Srinivasan and Mr. David B. Wendland for encouraging and allowing me to pursue and complete this endeavor.

To my wife Nila,

without whose push and unflinching support, none of this would have been possible.

To my daughters Aditi and Abha,

my daily sources of inspiration, who understood when daddy could not be with them on countless evenings and weekends during this project.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I: INTRODUCTION

1.1 INTRODUCTION

The application of medical devices in modern healthcare delivery is wide-ranging. Medical devices play a critical role in nearly all aspects of healthcare today, including prevention, diagnosis, therapy, monitoring and management.

Like in most industries, computers are deeply integrated into modern medical devices. From diagnostic laboratories to operating rooms, computerized medical devices are transforming healthcare delivery. For example, modern analyzers used in pathology laboratories can auto download test orders from the lab's information system, perform tests, validate results, and release reports to clinicians, often without any intervention from human operators. Similarly, old paper-based medical charts are increasingly replaced with electronic medical record systems. Today's operating rooms are fitted with sophisticated surgical robots that perform complex procedures on patients with guidance from surgeons. Ubiquitous wearable devices are collecting unprecedented amounts of clinical data from patients and consumers and opening new horizons for discoveries through advanced analytics.

The increased sophistication in medical devices also comes with an increased complexity and additional opportunities for failure. It is conceivable that computing technology-related medical device failures could have serious impact on patient health and

safety. Complexities around computing technology, such as software, are known to be particularly hard to measure and control. Traditional modes of controlling risks may not be entirely effective in mitigating against computing technology-related device failures.

Yet, there is a dearth of existing research on medical device safety in general, and particularly acute on the role computing technology plays. How frequent are computing technology-related medical device failures? Are computing technology-related medical device failures adequately reported? Do computing technology-related failures really cause severe harm to patients? We embarked on this research to find answers to some of these questions.

1.2 BACKGROUND

This study was conducted in multiple phases. The first phase was a preliminary exploratory research to assess if our initial questions had already been answered by existing research or by a publicly available dataset that could be easily analyzed. After an extensive review of the existing literature (discussed in greater detail in the Chapter II), we came to the conclusion that the existing body of research had not adequately answered our questions. However, we found a comprehensive dataset in the public domain that appeared promising enough to contain the information we were looking for.

After spending a significant amount of time and effort analyzing the data, we found that a critical field in the dataset that we used to identify computing technology-related medical device failures may have contained unreliable values. We then decided to abandon that approach and follow an alternative and much more complex approach for identifying computing technology-related medical device failures. This preliminary research and the discovery of a potential flaw in the publicly available dataset, however, led us to include a new hypothesis in our research.

In the sections below, we discuss in detail the initial approach we followed and the problems discovered. We believe that this background information will help the Reader understand the context behind our research hypotheses and the design of our subsequent experiments. However, a Reader not interested in the detailed context may skip to Section 1.3.

1.2.1 Preliminary Research

We realized early in our preliminary research that there is a severe lack of reliable existing literature on the topic of computing technology in medical devices, despite the abundance of anecdotal references and recognition of the central role computers and associated technologies increasingly play. However, we also discovered that there was a strong emphasis from the U.S. Food and Drug Administration (FDA) on the safety of medical device *software*. We found that the agency had not only published a number of guidance documents for the industry on software (FDA, 1999, 2002, 2005, 2014a, 2016a, 2016c, 2016e), it had also tracked and published information on medical device recalls caused by software (FDA, 2014b).

What we found to be the most relevant to our initial questions was the Manufacturer and User Facility Device Experience (MAUDE) dataset that the FDA has maintained as a part of its Medical Device Reporting mandate. This dataset contained millions of records dating back to 1990s on all reported incidents of medical device failures (FDA, 2017a). The dataset also included patient outcome and problem information (including computing technology-related problems) for the reported events. The MAUDE data also appeared to have been widely used in research in general (we provide a sampling of existing studies in Section 2.3.2) but studied negligibly from computing technology perspective.

The amount of potentially insightful information contained in the dataset made MAUDE an attractive source of data for our study. What followed was an analysis of the dataset and some preliminary observations that led to the formation of our hypotheses and an extensive study documented in this thesis. We discuss the methods and results of the preliminary research in the sections below.

1.2.1.1 Data Acquisition

We downloaded the MAUDE data files from the FDA's website on March 1, 2017 and imported them to a Microsoft SQL Server database. The need for the use of a sophisticated database system for data acquisition and analysis was due to the size of the raw data files in the dataset. Most files were too large to be opened in text editors or common office applications such as Microsoft Excel or Access.

Table 1 lists the entities created in the SQL Server database and populated with the relevant imported data:

Table	**Contained Data**	**Number of Records**
DEVICE_PROBLEM_CODES_ORIG	List of all current problem codes used by FDA in classifying failures.	988
FDA_CDRH_NCIT_SUBSETS_ORIG	Master vocabulary of all (current and historic) problem codes used by FDA. This vocabulary is published	2,003

Table	Contained Data	Number of Records
	by the National Cancer Institute.	
MDR_FOI_THRU_2016_ORIG	Master list of all reports of medical device failures from 1991 to 2016.	5,921,740
MDR_FOI_CHANGE_ORIG	All changes to the failures reports.	122,417
FOI_DEV_PROBLEM_ORIG	Mapping of problem codes to failures.	2,725,809
FOI_TEXT_2007_ORIG	Free-form narrative data on failures for the year 2007.	232,626
FOI_TEXT_2008_ORIG	Free-form narrative data on failures for the year 2008.	264,971
FOI_TEXT_2009_ORIG	Free-form narrative data on failures for the year 2009.	388,041
FOI_TEXT_2010_ORIG	Free-form narrative data on failures for the year 2010.	697,472

Table	Contained Data	Number of Records
FOI_TEXT_2011_ORIG	Free-form narrative data on failures for the year 2011.	972,480
FOI_TEXT_2012_ORIG	Free-form narrative data on failures for the year 2012.	1,251,520
FOI_TEXT_2013_ORIG	Free-form narrative data on failures for the year 2013.	1,536,462
FOI_TEXT_2014_ORIG	Free-form narrative data on failures for the year 2014.	1,965,058
FOI_TEXT_2015_ORIG	Free-form narrative data on failures for the year 2015.	2,274,087
FOI_TEXT_2016_ORIG	Free-form narrative data on failures for the year 2016.	2,106,765
FOI_TEXT_CHANGE_ORIG	All changes to the narrative data.	327,782
PATIENT_THRU_2016_ORIG	Patient treatment and outcome data associated with the reported events through 2016.	5,923,814

Table	Contained Data	Number of Records
PATIENT_CHANGE_ORIG	All changes to patient treatment and outcome data.	122,430

Table 1: Tables with Data Imported from MAUDE for Analysis

The master event report table (MDR_FOI_THRU_2016_ORIG) contained a total of 5,921,740 reports of medical device events since the FDA started keeping the record. Table 2 shows the yearly breakdown of these events:

Year	Reports
1991	15
1992	3,098
1993	4,408
1994	11,272
1995	9,758
1996	32,789
1997	77,691
1998	61,652
1999	52,909

Year	Reports
2000	52,570
2001	58,391
2002	69,595
2003	77,003
2004	81,805
2005	98,943
2006	119,640
2007	171,322
2008	194,424

Year	Reports
2009	241,895
2010	303,065
2011	445,118
2012	485,879
2013	679,224
2014	861,826
2015	861,045
2016	866,402
2017	1

Year	Reports
Total	**5,921,740**

Table 2: Yearly Breakdown of Event Records in the MAUDE Database

1.2.1.1.1 Computing Technology-Related Problem Codes

Next, we reviewed the FDA's problem's code dataset, which was also downloaded from the MAUDE portal and imported into the SQL Server database for analysis. The problem codes dataset contained a list of a total 988 different codes for functional deficiency. Events published in the MAUDE database are classified with none, one or more of these problems. Of these, we selected a subset of 32 codes as computing technology related using the FDA's Device Problem Code Hierarchy (DPCH), which contains a categorized list of problem codes. Most of these problems are listed under the Computer Software Issue category in the DPCH. Table 3 lists the problem codes we identified as computing technology related (see Appendix C for an excerpt from the DPCH):

Code	Description
1047	Failure to back-up
1048	Failure to convert to back-up
1110	Computer failure*
1111	Computer hardware error*
1112	Computer software issue

Code	Description
1138	Application interface becomes non-functional or program exits abnormally
1189	Dose calculation error due to software problem
1449	Parameter calculation error due to software problem
1473	Power calculation error due to software problem
1495	Incorrect software programming calculations
2581	Year 2000 (Y2K) related problem*
2582	Date-related software issue
2851	Date-related problem, year 2000 (Y2K)*
2879	Application network issue
2880	Application program issue
2881	Application program version or upgrade problem
2882	Application security issue
2898	Computer operating system issue
2899	Computer system security issue
2902	Data back-up problem
2903	Loss of Data
2963	Incorrect error code
2996	Operating system becomes non-functional
2997	Operating system version or upgrade problem
3013	Problem with software installation
3014	Programming issue

Code	Description
3025	Unauthorized access to computer system
3041	Computer Hardware**
3046	CPU (Central Processing Unit Of Computer System)**
3196	Data Issue
3197	Patient Data Issue
3198	Medication Error***

Table 3: Computing Technology-related Problem Codes

Some problem codes (those with * in Table 3) were not present in the DPCH, but present in the MAUDE list of problem codes. We included those in our preliminary research based on their similarity with the rest of the problem codes selected.

Some problem codes (those with ** in Table 3) were not present in either Device Problem Code Hierarchy or in the MAUDE list of problem codes. However, there were events in the MAUDE database associated with these codes. We found that these problem codes were defined in the Structured Product Labeling vocabulary, a collaboration between the FDA and the Enterprise Vocabulary Services of the National Cancer Institute (NCI, 2017).

One problem code, with *** in Table 3 (3198 - Medication Error), appeared not clearly computing technology related. However, the DPCH currently places this under Computer Software Issues (See Appendix C). Therefore, we opted to honor the DPCH and included it in our list.

1.2.1.2 Findings of Preliminary Research

1.2.1.2.1 Total Events

For the years of our interest (2007 - 2016), we found a total of 5,110,200 reports of medical device events in the MAUDE database. Table 4 shows the yearly breakdown of the number of reports:

Year	**Count**
2007	171,322
2008	194,424
2009	241,895
2010	303,065
2011	445,118
2012	485,879
2013	679,224
2014	861,826
2015	861,045
2016	866,402
Total	**5,110,200**

Table 4: Total Event Reports in MAUDE by Year

1.2.1.2.2 Computing Technology-related Events

Using the computing technology-related problem codes listed in the FDA's Device Problem Code Hierarchy (DPCH) as a guide, we then queried the MAUDE database for the

number of computer technology-related events. We found that a total of 20,224 events were related to computing technology. Table 5 provides a yearly breakdown of these events:

Year	**Count**
2007	9,766
2008	907
2009	740
2010	431
2011	878
2012	548
2013	276
2014	676
2015	2,721
2016	3,301
Total:	**20,244**

Table 5: Total Computing Technology-related Event Reports by Year

1.2.1.2.3 Portion of Computer technology-related Events

Based on the problem codes assigned in MAUDE, we found that only 20,244 or 0.4% of the total 5,110,200 reports of medical device events were related to computing technology. Table 6 provides a yearly breakdown of the portion of events related to computing technology:

Year	Total Events	Computing Technology-related Events	Percentage
2007	171,322	9,766	5.7%
2008	194,424	907	0.47%
2009	241,895	740	0.31%
2010	303,065	431	0.14%
2011	445,118	878	0.2%
2012	485,879	548	0.11%
2013	679,224	276	0.04%
2014	861,826	676	0.08%
2015	861,045	2,721	0.32%
2016	866,402	3,301	0.38%
Total	**5,110,200**	20,244	0.4%

Table 6: Percentage of Computing Technology-related Medical Device Events using FDA's Problem Codes

1.2.1.2.4 Trend

Medical device failures reported to the FDA in the years 2007 - 2016 and published on the MAUDE database show a steady rise followed by a plateau in recent years. The annual number of failures reported grew more than 500% in the study period, from 171,322 in 2007 to 866,402 in 2016. Overall, the growth demonstrates a strong linear trend, as illustrated in Figure 1:

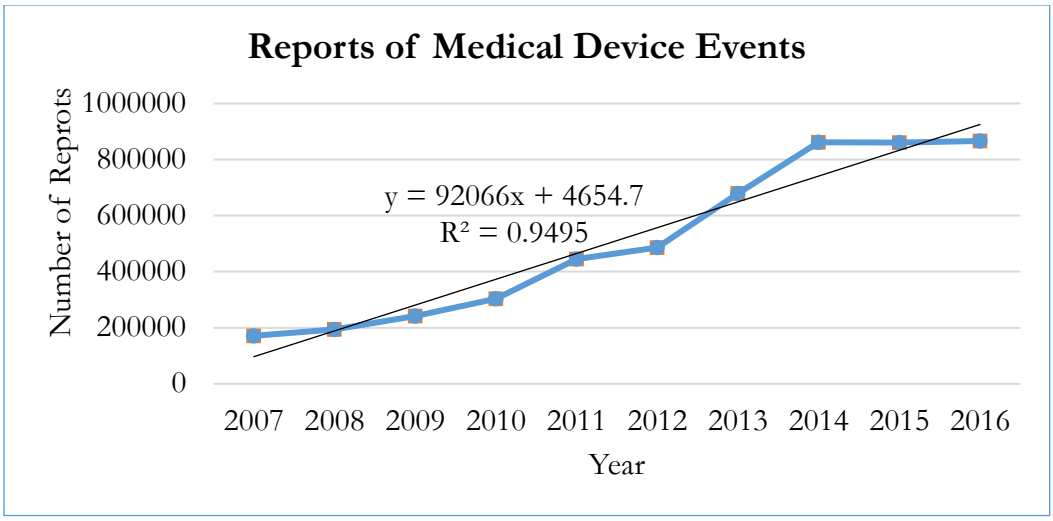


Figure 1: Trend of Medical Device Failure Events

Unlike the number of overall failures, events related to computing technology, however, do not show a clear trend for the study period based on the problem code assignment in MAUDE. Figure 2 shows the yearly total number of reports of medical device events related to computing technology:

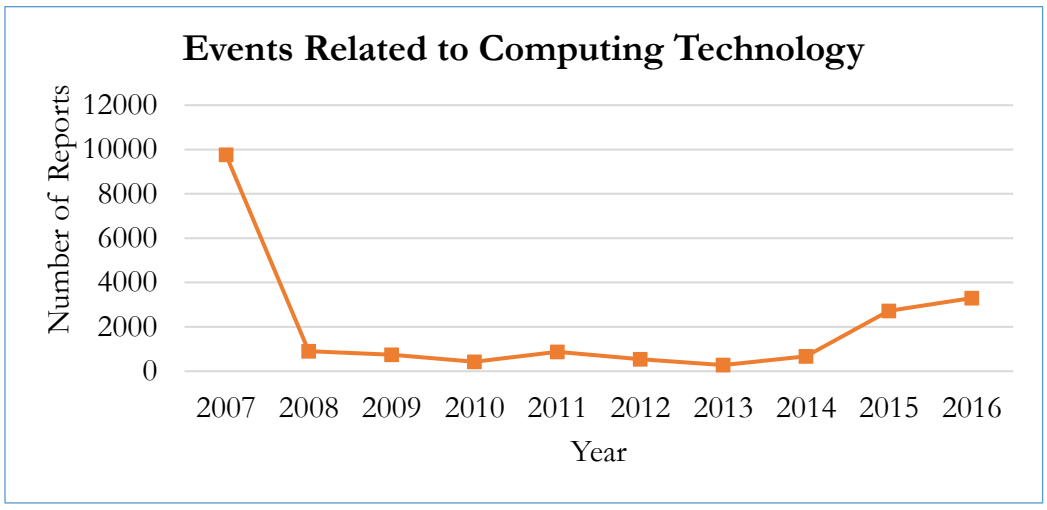


Figure 2: Trend of Medical Device Events Related to Computing Technology per FDA's Problem Codes

The lack of clear trend in the computing technology-related medical device events based on problem codes is also evident from the proportion of these events to the overall number of events. While the overall number of events have maintained a linear growth, the annual percentage of reports of events related to computing technology, however, have not shown any direction as illustrated in Figure 3:

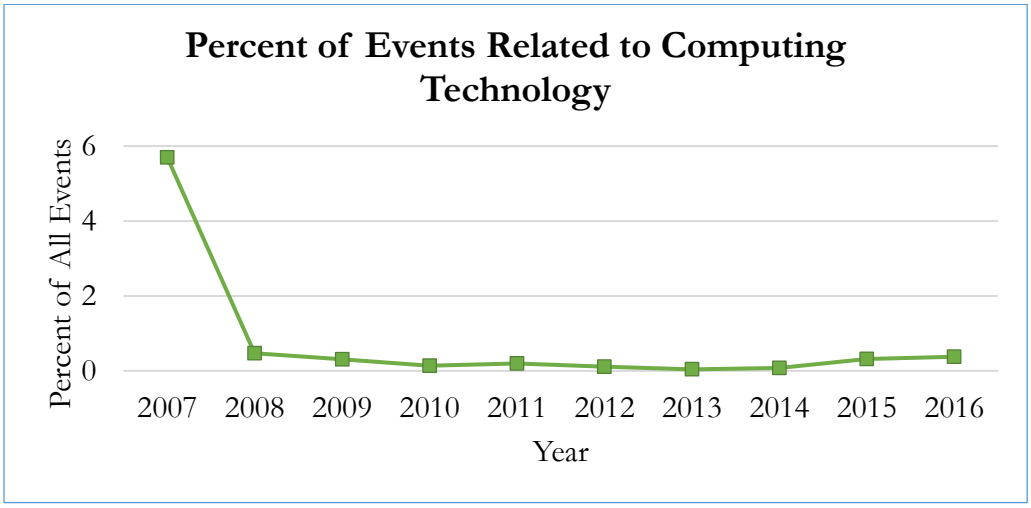


Figure 3: Percent of Medical Device Events Related to Computing Technology per FDA's Problem Codes

1.2.1.2.5 The 2007 Issue

To explain the 2007 anomaly, we investigated the cause for the extraordinarily high number of reports. We found that one manufacturer alone submitted 9,486 (97.13% of all reports of computing technology-related events submitted that year) reports of medical device failure across just two problem codes (Loss of Data, Computer operating system issue).

1.2.1.3 Problems

The findings of the preliminary research discussed in the previous sections exhibited two major problems. First, the reports of computing technology-related medical device events were insignificant (less than 0.5% of total reports) for all years except 2007. This was contrary to our initial assumption and professional experience that computing technology played a central role in modern medical devices. The second problem was that, while the increase in the overall number of failures from 2007 to 2016 showed a strong linear trend, computing technology-related failures showed no such trend for the same period.

These problems led us to believe that the problem codes assigned to medical device failures in the MAUDE datasets published by the FDA may not be accurate for computing technology-related events. While the FDA does not state how, if at all, the agency curates the MAUDE data before it is published, our preliminary findings seemed to suggest that the current problem code assignment scheme may be susceptible to the submitter's interest and ability to navigate the FDA's Device Problem Codes Hierarchy (DPCH).

Therefore, we decided that we could not rely on the problem code assignment in the MAUDE database to identify a report of medical device failure as potentially related to computing technology. However, without a more reliable method to identify computing technology-related events, we could not answer our initial questions. This predicament led us to purse an alternative method for identifying computing-technology failures, which became a major component of the research documented in this thesis.

1.2.2 Narratives

In addition to the structured data that we mapped to a set of relational database tables for our analysis, the MAUDE dataset we downloaded also included a set of large text files

containing natural-language narratives describing the reported events. In a random manual sampling of these narratives, we found that they contained information that could potentially be used to identify failures that are computing technology-related. We also discovered from our sampling that some clearly computing technology-related failures based on the narratives had been classified in MAUDE with either inaccurate or ambiguous problem codes.

Considering the information value contained, we decided to utilize the over 11 million records of narratives in the MAUDE dataset to identify computing technology-related medical device failure events. Our first goal was to classify each of the narrative records as either computing technology-related (positive) or not related (negative) based on the natural-language description in the narrative. Then, using the results of the classification and other related data in MAUDE, we sought to analyze the reports of medical device failures around our research hypotheses.

1.3 RESEARCH HYPOTHESES

Based on our initial set of research questions, the availability of the MAUDE data for analysis and the findings of our preliminary research, we formed four primary hypotheses for our study:

1. Limited existing literature and the results of our preliminary research show that only negligible portion of medical device failures are related to computing technology. However, we believe that these results do not accurately represent the reality where we find computers and the associated technology deeply integrated into a wide range of medical devices. Our expectation is that this pervasiveness would be reasonably reflected in the number and proportion of computing technology-related medical device failures reported to the FDA.

Thus, we hypothesize that *computing technology is related to a significant portion of medical device failures reported to the FDA.*

We recognize the ambiguity associated with the term '*significant portion*' in this hypothesis. We will arbitrarily consider this term to mean *at least five percent* for the purpose of our research.

2. Considering the ubiquitous integration of computing technology across industries and the proliferation of health-related mobile applications and "smart" devices, we believe that computing technology is *increasingly* playing a greater role in medical devices. We extend this observation to hypothesize that *computing technology is related to an increasing number of medical device failures reported to the FDA.*

3. It is known that medical device failures can have fatal consequences on patients. What is not known is whether computing technology-related medical device failures contribute to such consequences. Assuming that computing technology plays a central role in medical devices, we hypothesize that *medical device failures related to computing technology can have fatal consequences on patients.*

4. The findings of our preliminary research suggested that the current scheme of assigning problem codes to medical device failures in the FDA's Medical Device Reporting program may be problematic. In particular, we hypothesize that *the problem codes assigned to medical device failures in the FDA's MAUDE database are masking computer technology-related problems.*

1.4 SIGNIFICANCE OF THE STUDY

We believe that this study contributes to the existing body of research on our understanding of medical devices and their reliability in several significant ways:

First, despite the prevalence of computing technology, very limited existing research exists on the role it plays on medical devices. In particular, virtually no reliable information is currently available on medical device failures related to the onboard or dependent computing technology and the impact of such failures on patients. This research provides a baseline for future work on this topic.

Secondly, the MAUDE dataset, although very comprehensive with millions of records, has not been utilized to its potential by the research community. All existing studies based on MAUDE data we have found were conducted using the search engine on the FDA's Website[1], which queries the MAUDE database and returns paged results based on specific search parameters. Our study, however, is performed using the raw data files that the FDA also publishes[2], giving us significantly greater access and control over the data and the analyses that could be performed. No other study, as far as we could find, has examined the MAUDE dataset, and particularly the natural language narratives at this magnitude before. We believe that this study will encourage more researchers to examine the raw files in the MAUDE dataset for new discoveries and insights.

Third, we have discovered through our preliminary research that there may be significant flaws in the current scheme of assigning problem codes in the data published by

[1] https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfmaude/search.cfm

[2] https://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/PostmarketRequirements/ReportingAdverseEvents/ucm127891.htm

the FDA through the MAUDE database. In particular, we have theorized that the problem codes assigned to events in the MAUDE database are masking computing technology-related problems. A definitive investigation on this theory could help determine if any corrective actions are necessary on the part of the FDA, the industry or the research community currently relying on the assigned problem codes.

Last but not least, we have sought, in this research, to utilize machine learning techniques to identify medical device failures related to computing technology, likely making it the first study to do so. We believe that this study will encourage further research on the application of machine learning and artificial intelligence in reducing medical devices failures in the future.

1.5 RESEARCH OBJECTIVES

The overall purpose of this research is to help improve the quality of healthcare through safer and more reliable medical devices. In this study, we seek to analyze medical device failures reported to the FDA over the period of 10 years (2007 - 2016) and shed some light on the current state, gaps and opportunities for improvement toward this purpose. More specifically, our research has the following objectives:

1. Conduct a comprehensive literature review on the state of computing technology in medical devices and their failures;
2. Conduct a comprehensive literature review on the state of machine learning algorithms and techniques with a particular focus on classification problems;
3. Create machine learning models to identify medical device failures related to computing technology;

4. Using the machine learning models created, classify each report of medical device failure published by the FDA in the MAUDE dataset as either computing technology related or not related; and
5. Analyze all identified reports of computing technology-related medical device failures against the four research hypotheses described in Section 1.3.
6. Discuss what actions can be taken based on the findings of this research to improve the safety and reliability of medical devices.

CHAPTER II: REVIEW OF LITERATURE

2.1 INTRODUCTION

Given the wide-ranging applications in healthcare, the definition of the term *medical device* is broad. From fairly simple tools such as thermometers to highly sophisticated systems such as surgical robots, just about any instrument or system used for the purpose of prevention, diagnosis, treatment or management of a disease condition can be considered a medical device.

Computing is a similarly broad discipline that includes several subfields of study such as computer engineering, information systems, information technology and software engineering. Computing technology can thus be considered any component or accessory that plays a role in the performance of a computer system. Examples of computing technology include processors, monitors, keyboards, disks, software, firmware, networking and digital communication.

As medical devices become more complex, they also become more prone to failures. Despite the increasingly greater role computing technology plays, our knowledge of medical device failures related to computing technology is limited. In a recent report, the U.S. Food and Drug Administration (FDA) attributed 15% of all medical device recalls to software-related causes. However, this information is based on recalls only, and may not truly reflect

the rate of medical device failures. Some other studies have also indicated significant issues with health information technology, but no reliable information exists on medical device failures related to computing technology.

The Manufacturer and User Facility Device Experience (MAUDE) database is a component of the FDA's federal mandate on post-market medical device surveillance in the United States. This database is a central repository of all reports of medical device failures the FDA has received dating back to 1990s. For each report of failure, the MAUDE database contains information on the event, device, patient outcome, causes of the failure, and a set of textual narratives.

There have been several studies on medical device failure using the MAUDE database. Most of these studies have been focused on devices or technology in the context of a specific clinical specialty. A very limited set of studies have utilized the MAUDE database to examine device failures related to computing technology.

Machine learning is a discipline in artificial intelligence concerned with allowing software programs to make decisions through patterns in the data, rather than traditional methods of explicitly-coded instructions. In supervised learning, the computer is provided with training data to 'learn' significant patterns present, which it then uses to make decisions on previously unseen data. In unsupervised learning, the computer is tasked with identifying structures in the data without any samples to train on.

One of the foundational applications of machine learning has been in information extraction from text-based sources. Named entity recognition is a process of identifying real-world objects from textual data. Text classification is a process of assigning predefined labels

to records based on the presence of features of interest. Relationship extraction is a process of characterizing semantic relations among entities in textual data.

Several algorithms have been defined for classification-related machine learning tasks. In the naïve Bayes approach, the classification is based on the probability of an event given known prior probabilities in the training data. In linear regression, the classification is based on the fitness of the unseen data against the linearly separable clusters in the training data. Logistic regression is a classification technique that solves the problems of potentially out-of-range probability values when a linear model is applied in a classification task involving dichotomous classes. Support vector machine allows the margins between two linearly separable classes in the training data to be maximized. Gradient descent is an optimization method that iteratively minimizes the loss in a model's predictions.

In the sections below, we provide a more detailed definition and analysis of the current state of research on each of the topics discussed in this overview.

2.2 MEDICAL DEVICES AND COMPUTING TECHNOLOGY

2.2.1 Definitions

2.2.1.1 Medical Device

Given the variety of applications, the definition of a medical device is generally broad. The FDA defines a medical device as (FDA, 2016f):

> *An instrument, apparatus, implement, machine, contrivance, implant, in vitro reagent, or other similar or related article, including a component part, or accessory which is:*

- *recognized in the official National Formulary, or the United States Pharmacopoeia, or any supplement to them,*
- *intended for use in the diagnosis of disease or other conditions, or in the cure, mitigation, treatment, or prevention of disease, in man or other animals, or*
- *intended to affect the structure or any function of the body of man or other animals, and which does not achieve any of its primary intended purposes through chemical action within or on the body of man or other animals and which is not dependent upon being metabolized for the achievement of any of its primary intended purposes*

In addition to this overarching definition, the FDA has established a classification system to categorize medical devices based on their potential impact on patient health and safety. The level of regulatory oversight a medical device receives and its pre-marketing requirements generally depend on this classification. In the current system, the FDA classifies each medical device into one of the categories listed in Table 7 (FDA, 2015):

Class	**Description**
Class I	This class of medical devices are deemed to pose the lowest risk to the patient or the User and are subject to the least regulatory controls.
Class II	Medical devices in this class pose higher risk than Class I devices and require manufacturers to provide reasonable reassurance of their safety and effectiveness.
Class III	Medical devices in class have the highest risk to the patient or the User and are subject to elaborate regulatory controls, even before they are marketed.

Table 7: FDA's Medical Device Classification

Similar classification systems exist in many other jurisdictions, as well. The determination of which medical device belongs to which regulatory class is a complex process and includes considerations for intended purpose, invasiveness, scope and duration of application.

For the purpose of this research, we adopt the FDA's definition of medical device as excerpted above.

2.2.1.2 Computing Technology

The Association of Computing Machinery (ACM) defines *computing* as a *goal-oriented activity requiring, benefiting from or creating computers* (Shackelford et al., 2006). The ACM considers computing a broad discipline that includes *computer engineering*, *computer science*, *information systems*, *information technology* and *software engineering*. However, there is little consensus in either the academia or industry on exactly where the boundaries lie for many of these sub-disciplines. For example, some consider information technology (IT) as an umbrella term that encompasses everything computing technology-related, while others view IT as a field concerned with only the application and administration of computing artifacts in an enterprise. Similar ambiguities also exist between computer science and software engineering. We consider this variety in definitions immaterial to our research and find the ACM's overarching definition of computing to be appropriate for our study. Leveraging this definition, we consider, for the purpose of this research, computing technology simply as:

Any system, application, device, component or accessory, software or hardware, that plays a contributing role in the performance of a computer.

We define *computer* in this context as a *programmable electronic device that can accept, store, retrieve, process, display or transmit data*. This definition is built upon a more basic definition

found in the dictionary: *a programmable usually electronic device that can store, retrieve, and process data* (Merriam-Webster, 2017).

Based on these definitions, we present below some examples of computing technology:

- Computers
- Accessories (keyboards/keypads, mouse/joysticks, display monitors, disks/removable media)
- Integrated circuit boards, storage (volatile or persistent), digital sensors, meters, transmitters, receivers
- Microcontrollers, microprocessors
- Software, firmware
- Internet, networking apparatus
- Electronic files, records
- Wired or wireless digital communication

2.2.2 Role of Computing Technology in Medical Devices

The adoption of computers and computing technology is ubiquitous in today's medical devices. While no reliable statistics exists on exactly what portion of medical devices are computer-driven, there is evidence that computing technology increasingly plays a greater role in medical devices (Fu, 2011). Advancements in computing and related technologies have facilitated proliferation of medical devices, which are becoming smaller in their physical size but bigger in terms of their software footprint (Lee et al., 2006).

Medical devices are a key component of the modern healthcare delivery system. As they become more and more computerized and integrated with the enterprise infrastructure, computing technology is becoming a critical component at every stage in the healthcare delivery process, from primary to critical care. A patient in an intensive care unit of a hospital can have multiple computerized devices attached to monitor the physiology and deliver therapy. These include devices such as an ECG monitor, blood pressure monitor, pulse oximeter, intravenous pump and a ventilator, all connected to a bedside patient monitor (Gardner, Clemmer, Evans, & Mark, 2014).

Medical imaging is another area where computers and software have revolutionized the options available to clinicians. From anatomic and functional imaging to structural modeling of nucleic acids and proteins, measurement of gene expression (e.g. through fluorescence), morphometrics and visual aid, computerized imaging devices and analyzers play an important role in pathology, radiology and surgery (Erickson & Greenes, 2014; Rubin, Greenspan, & Brinkley, 2014).

Even implantable medical devices such as cardioverter defibrillators, neurostimulators, pacemakers and drug pumps use embedded computers (Halperin et al., 2008). Computerized assistive devices and medical robots are also used in surgery (Taylor, Menciassi, Fichtinger, & Dario, 2008). Electronic health records and associated tools such as clinician order entry and clinical decision support are sophisticated computerized systems (McDonald, Tang, & Hripcsak, 2014).

Computer, software and communication technologies power devices used in telemedicine. This includes remote interpretation of diagnostic test results, telehealth and remote intensive care (Starren, Nesbitt, & Chiang, 2014). A variety of software-driven tools

and applications have also been in use in patient engagement, communication and even behavior management (Johnson, Jimison, & Mandl, 2014). A new class of wearable devices (smart watches, glasses, activity trackers) is emerging with clinical and health-related applications (Boulos, Brewer, Karimkhani, Buller, & Dellavalle, 2014).

Laboratory medicine is also undergoing transformation as computing technology gets deeply integrated into the lab workflows. Diagnostic devices in today's laboratory contain robotic sample handling, and connectivity to the lab's information systems. This has enabled a complete automation of specimen testing for most tests from sample preparation to result release (Jones, Johnson, & Batstone, 2014).

2.2.3 Mobile Health and Software as a Medical Device

Systems such as electronic health records (EHR) and clinical decision support systems (CDSS) have challenged the traditional understanding of a medical device as a primarily physical, hardware apparatus. However, the question of whether such health information technology (HIT) systems can be considered medical devices is a contentious one -- mainly due to the regulatory implications of the argument. Some consider the current situation (i.e. "HIT is not a medical device") a fallacy enabling the use of safety-critical systems in healthcare delivery without any independent assessment of their safety or fitness (Karsh, Weinger, Abbott, & Wears, 2010).

The mobile revolution of recent years has amplified the notion of software as a medical device in an unprecedented way: Thousands of health-related applications are now available to consumers with a smartphone, a highly portable and personalized computer with messaging, imaging (e.g. still photo, video) location (e.g. global positioning system or GPS) and other sensing capabilities (Boulos et al., 2014).

A systematic review of healthcare applications for smartphones found that many applications were available for healthcare professionals, medical or nursing students, and for patients. For healthcare professionals, applications were available for disease diagnosis, drug reference, medical calculations, literature search, hospital information systems connectivity and medical training. Similarly, for medical and nursing students, applications were available for surgery notebook, eponyms, atlas of human anatomy, dissection, cranial nerves and electrocardiography. Applications were also available for patients on chronic disease management (e.g. bolus insulin dose calculation, asthma peak-flow monitoring, cardiac rehabilitation, pulmonary rehabilitation, sleep aid and sound therapy) and assessments such as fall detection, hearing check, and physical activity level measurement (Mosa, Yoo, & Sheets, 2012). Applications are also available on epidemiology (e.g. outbreak detection), behavioral health, and on the management of patients with psychiatric conditions and dementia (Boulos et al., 2014).

2.2.4 Medical Device Failures

Despite the magnitude of medical devices in use in healthcare today, limited pre-existing, objective literature exists on their failure. This was highlighted by (Ward & Clarkson, 2004) and while some progress has been made in terms of the availability of data and specialty-focused studies, the topic remains largely ignored by researchers.

In the United States, the FDA collects reports of medical device failure through Medical Device Reporting (MDR), a set of provisions under the federal regulation 21 CFR 803 (FDA, 2017b). As a part of the MDR, manufacturers and importers are required to report medical device-related failures and product issues to the FDA. Device user facilities, such as hospitals or ambulatory clinics are also required to report medical device related

adverse patient events (such as death or injury) to the FDA. The MDR also allows and encourages healthcare professionals, patients, caregivers and consumers to report significant failures or problems voluntarily. The submissions to the FDA are made through the MedWach Form FDA 3500 for voluntary reports and Form FDA 3500A for mandatory reports. The forms are available in paper (see Appendix A and B), online or as a smartphone application (FDA, 2016b).

The data the FDA collects as a part of the MDR is made available to the public through an online search interface and in the form of downloadable raw files from the Manufacturer and User Facility Device Experience (MAUDE) Database (FDA, 2017a). This database will be the subject of a closer examination and analysis in later chapters in this thesis.

The FDA also keeps a record of all medical device recalls. A recall is when a company takes a correction or removal action to address a problem with their medical device (FDA, 2017c). A medical device recall falls into one of three classes based on the relative degree of risk:

- Class I (where the medical device has a reasonable chance of causing serious health problems or death);
- Class II (where the medical device may cause temporary health problem, or slight chance of causing serious health problems or death); and
- Class III (where the medical device is not likely to cause any health problem or injury).

In most cases, marketers of medical devices voluntarily initiate the recalls, although the FDA can also require a company to recall a device. Manufacturers are required to notify the

FDA of all recalls, which are then posted on the Medical Device Recall Database and published to the public (FDA, 2017c).

The FDA has on occasions published an analysis of the recall data. Perhaps the most significant of such analyses is the Medical Device Recall Report released in 2014 analyzing medical device recalls over a 10-year period from fiscal year 2003 to 2012 (FDA, 2014b). This analysis found that overall annual recall counts increased 97% from 603 recalls in 2003 to 1,190 in 2012. Highest severity (Class I) recalls grew from 1% of all recalls in 2003 to 5% of all recalls in 2012 and most frequently involved infusion pumps, automated external defibrillators, continuous ventilators, over-the-counter blood glucose test systems, catheter introducers, and implanted infusion pumps. In many cases, Class I recalls were a result of fatal incidents. Death was associated with 25% of the Class I recalls. Most of these recalls were for devices used in the general hospital, followed by cardiovascular, anesthesiology, chemistry, gastrology, urology, microbiology general surgery, radiology, neurology and ophthalmic specialties.

The FDA analysis also found that a small subset (0.15%) of the product types were responsible for 20% of all medical device recalls. These included medical linear accelerators, radiological image processing systems, infusion pumps, computed tomography x-ray systems, automated external defibrillators, electrosurgical accessories, chemistry analyzers, chemistry data processing modules, cell differential counters and semi-constrained patello-femorotibial knee prosthesis.

The rate of medical device recalls appears to be related to the jurisdiction in which they are first marketed suggesting a variability in the level of expected pre-market quality across jurisdictions. (Hwang, Sokolov, Franklin, & Kesselheim, 2016) studied the recall rates

of medical devices launched in the European Union and the United States. They found that 63% of devices that are approved in both the EU and the US were first approved in the EU. They also found that the recall rate of devices first approved in the EU was higher (27%) than the recall rate for devices first approved in the US (14%). As a context, medical devices in the EU are approved for marketing (such as through a Conformité Européenne, or CE mark) by private notified bodies. This is in contrast with the approval process enforced by the FDA, a powerful government agency, in the United States.

However, the effectiveness of the FDA's approval process as a safeguard to ensure medical device safety and quality has also been called into question. The FDA has two approval pathways for medical devices:

1. The Pre-market Approval (PMA) process, a highly rigorous pathway generally applicable to new or safety-critical devices; and
2. The 510(k) process, which is considered less rigorous and applicable to devices with the type ("predicate device") that already exists on market.

(Zuckerman, Brown, & Nissen, 2011) studied if the FDA's premarket classification to determine the approval pathway for a medical device was effective and reflective of the safety risks contained in the device. They analyzed 113 highest-risk recalls of medical devices by the FDA attributable to the risk of causing serious health problems or death, also known as Class I recall. They found that only 21 (19%) of the recalled devices were approved through the PMA process. Rest of the Class I recalled devices went through the less rigorous 510(k) process (71%), were exempted from the FDA approval process altogether (7%) or were counterfeit products (3%).

2.2.5 Computing Technology Failures in Medical Devices

Limited literature is available on computing technology failures in medical devices. In likely first and only comprehensive analysis on the topic to date, (Wallace & Kuhn, 2001) studied medical device recalls from 1983 to 1991 (15 years) and found that 6% of 2,792 medical device recalls in the study period were related to computer software. Of the software recalls, the faults were mostly in logic (43%), calculation (24%), change impact (6%), data (5%) and requirements (4%). Other classes of faults resulting in the recall included omission, timing, quality assurance, initialization, interface, and configuration management. The software recalls were mostly in devices used in radiology, followed by cardiology, diagnostic, anesthesiology, general hospital and surgery (Wallace & Kuhn, 2001).

(Alemzadeh, Iyer, Kalbarczyk, & Raman, 2013) studied 5,294 medical device recalls between 2006 and 2011 and observed that computer-related recalls contributed to 1,210 (23%) of all recalls in the period. Of the computer-related recalls, 94% presented some risk of a serious injury or even death. 778 (64%) of the computer-related recalls were related to software faults.

The FDA's analysis of 10 years (2003 - 2012) of medical device recall data discussed in the previous section also showed that 15% of the recalls between the fiscal years 2008-2012 were due to software-related causes (FDA, 2014b). This is a marked increase from the 6% found by (Wallace & Kuhn, 2001) in their study of the recall data from 1983-1991 and may provide further evidence of increasing role software plays in medical devices.

The FDA's analysis identified medical linear accelerator as the most frequently recalled type of product. Further analysis of the causes showed that software issues contributed to more than two-thirds of the accelerator recalls. The specific software issues included system

compatibility, interoperability, human factors, and dose calculation. For the fiscal years 2010-2012, *device software design issue* and *nonconforming material/component* were the top reasons for all medical device recalls. From fiscal years 2008 to 2012, software was the primary cause of the majority of medical device recalls in radiology, cardiovascular, clinical chemistry specialties, as well as in the general hospital (FDA, 2014b).

Some studies also exist on health information technology failure. (Magrabi, Ong, Runciman, & Coiera, 2011) found that from January 2008 to July 2010, there were 678 reports of 436 different adverse medical device events associated with health information technology. 46 (11%) of these events were associated with patient harm. Most of the 46 events were related to the computerized physician order entry (CPOE) (63%) and picture archiving and communication (PACS) (30%) systems (Magrabi et al., 2011).

(Meeks et al., 2014) studied 100 closed investigations of reported EHR-related safety concerns from 55 US Veterans Health Administration facilities between 2009 and 2013. They found that 74 were related to unsafe technology; 25 related to unsafe use of technology; and 1 related to the lack of monitoring of safety concerns. The EHR safety concerns included unmet display needs (mismatch between information needs and content display), software modification or upgrades (concerns due to upgrades, modifications or configuration), data transmission (failure of interface between EHR system or components), and hidden dependencies (one component of EHR is unexpectedly or unknowingly affected by the state of another component) (Meeks et al., 2014).

(Menon et al., 2016) conducted a retrospective analysis of one-year worth of notes from 'safety huddle', a 15-minute meeting every workday on safety attended by representatives from clinical, information technology and administrative departments at a

tertiary-care US hospital between 2013 and 2014. They found that of total 3,270 safety concerns, 245 (7%) were related to the electronic health record (EHR) system. The key EHR-related safety concerns included software malfunction (29% of all EHR-related safety concerns), incorrect or inaccurate clinical reference information (19%), human factors (13%), workflow mismatch (12%) and data display errors (7%) (Menon et al., 2016).

2.2.6 Challenges to Medical Device Safety and Reliability

Underreporting of medical device failures has been widely suspected (Rajan, Kramer, & Kesselheim, 2015; Ward & Clarkson, 2004). Even among reported incidents, the classification of their causes may not be accurate. For example, (Magrabi, Ong, Runciman, & Coiera, 2012) found that only 0.1% of nearly 900,000 medical device events recorded in the FDA's MAUDE database were health information technology related, which is at odds with the prevalence of computers and software in devices, discussed previously. The under-reporting may be, among other things, due to the limitations of the reporting systems or just a low expectation of reliability the users have with computers and computing technology (Magrabi et al., 2012). The lack of device identifiers has been noted as an impediment to safety monitoring of medical devices and the FDA's recent initiatives to require Unique Device Identifiers (UDI) on devices are expected to be helpful (Resnic & Normand, 2012; Rising & Moscovitch, 2014).

From regulatory perspective, the FDA's current processes for medical device approval and post-market surveillance appear to have some gaps. The pre-market classification of a medical device that governs its approval process does not necessarily correspond in practice with the safety risks it contains (Zuckerman et al., 2011). The agency's post-market surveillance of devices may also be lacking effectiveness. One study found that of 223 post-

approval studies (PAS) of 158 medical devices the FDA ordered between 2005 and 2011, only one device resulted in getting removed from the market (labeling change was requested in 31 cases). There were no instances of a warning letter issued or a penalty levied against a device manufacturer owing to a PAS finding (Reynolds, Rising, Coukell, Paulson, & Redberg, 2014). In at least one case, the FDA failed to recall a device despite PAS data suggesting significant safety risks, including mortality (Rajan et al., 2015).

Insufficient considerations for human factors in the design of medical devices has also been cited as a reason for medical failures. Often times, failures are categorized as 'user error', effectively putting the blame on the user, where in fact, the underlying cause is really a poor device design (Ward & Clarkson, 2004).

There are also challenges to medical device software. Ensuring safety remains among the top challenges, and more pressing as the complexity of the embedded software increases (Rakitin, 2006). Despite the potentially mission-critical role, medical device software is not necessarily developed using a well-established development process (Denger, Feldmann, Host, Lindholm, & Shull, 2007). Issues such as the inability of commercial off-the-shelf technologies to provide security, privacy, reliability and interoperability needed in a medical device; gaps in critical systems infrastructure; and lack of holistic approach that includes functional, computational and communication elements in embedded system designs; and limits of the validation and certification processes pose a challenge in the development of high-confidence medical device software and systems (Lee et al., 2006).

Some have suggested that application of formal, model-based methods in all phases of system development could be useful in ensuring the efficacy and safety of critical medical device software (Z. Jiang, Abbas, Jang, & Mangharam, 2016). This, however, may not be

practical. While there have been instances of successful use of formal methods in the verification of some safety-critical systems, formal methods in software engineering remain a controversial topic and only sparsely adopted (Clarke & Wing, 1996; Woodcock, Larsen, Bicarregui, & Fitzgerald, 2009).

Challenges also exist for health information technology (HIT), the primarily software systems used in healthcare delivery today. (Karsh et al., 2010) have identified 12 fallacies of current design and understanding of HIT. They include: Risk Free HIT Fallacy (HIT risks are minor and easily manageable); HIT is Not a Device Fallacy (HIT systems are not medical devices); Learned Intermediary Fallacy (the user, not HIT, is the one making the decision); Bad Apple Fallacy (users who won't use the system are just uncooperative); Use Equals Success Fallacy (if the system is used, its design is a success); Messy Desk Fallacy (HIT can enforce linearity in the complex and messy clinical workflows); Father Knows Best Fallacy (the system is good if it serves upstream stakeholders even if it does not aid the front-line worker); Field of Dreams Fallacy (the system is designed correctly, any evidence otherwise is user error); Sit-Stay Fallacy (clinicians should rely on HIT because computers are smarter than humans); One Size Fits All Fallacy (the system can be designed for isolated single-user sessions performing discrete tasks); We Computerized the Paper, So We Can Go Paperless Fallacy (HIT can eliminate paper-based artifacts); and No One Else Understands Healthcare Fallacy (no one outside the healthcare domain could possibly solve complex HIT issues) (Karsh et al., 2010).

For mobile medical applications, a major challenge is to ensure their efficacy and safety considering the sheer number of these "apps" available today. Some studies have shown that many of these applications are developed without any involvement of a medical

professional. The diagnostic accuracy of some of these apps has also been called into question (Boulos et al., 2014). The FDA has responded to some of these concerns by publishing guidelines and its intention on regulating a subset of these software-only medical devices (FDA, 2016d).

Some have questioned if financial penalties associated with medical device failures are not significant enough to cause manufacturers to require higher quality. To answer this question, (Thirumalai & Sinha, 2011) studied the impact of a medical device recall to the manufacturer's financial performance. They selected a sample of medical device recalls between 2002 and 2005 and assessed if a device recall had any negative impact on the manufacturer's shareholder wealth as reflected in the capital markets. The study found no significant evidence of such impact. However, it found that companies with larger product scope have higher likelihood of product failures. Similarly, a company's prior recall experience indicated a decrease in the likelihood of future recalls, suggesting a learning effect (Thirumalai & Sinha, 2011).

There are also arguments that medical device software is not transparent enough to patients, who end up paying the price (in some cases, in terms of life) for the defects. Some contend that making medical device software source code, particularly for safety-critical implanted devices, open and subject to public scrutiny and improvements could contribute to the overall safety and security of the devices (Sandler, Ohrstrom, Moy, & McVay, 2010).

2.3 MANUFACTURER AND USER FACILITY DEVICE EXPERIENCE DATABASE

2.3.1 What is MAUDE?

The Medical Device Reporting (MDR) is a component of the FDA's post-market surveillance mandate codified through 21-CFR-803 (FDA, 2017b). As a part of the MDR, the FDA requires manufacturers and importers of medical devices to submit to the agency any reports of failures related to their devices. Healthcare facilities such as hospitals are also required to report any mortalities related to medical devices. Also, the general public is encouraged to voluntarily report any medical device issues (FDA, 2016b).

The Manufacturer and User Facility Device Experience (MAUDE) is a central repository of reports of medical device failures published by the FDA based on the information it collects from medical device manufacturers, importers, healthcare facilities and the general public through the MDR. The MAUDE database contains information on medical device issues reported since the 1990s (FDA, 2017a).

2.3.1.1 Data Attributes

The FDA collects information published in the MAUDE through the FDA MedWatch program. Three types of forms are currently available for reporting. The FDA Form 3500 and its consumer-friendly version (FDA Form 3500B) can be used for voluntary submission by the public. Manufacturers, distributors and healthcare facilities must use the FDA Form 3500A for their mandatory submission. The MedWatch forms are available in paper form (see Appendix A and B) as well as online and as mobile applications. These forms capture a variety of data attributes in each report and a subset of these are published in the MAUDE database. We discuss some below:

2.3.1.2 Event Information

The MAUDE database contains basic information on each event reported. The information includes, date/time of the event, reporting party (e.g. voluntary, manufacturer, distributor, user facility, etc.), reporter occupation, location of the event, problem code(s), number of devices impacted by the event, number of patients impacted by the event, etc.

2.3.1.3 Device Information

For each medical device event reported, the MAUDE database contains information identifying the device. This includes, device catalog number, brand name, common name, model number, lot number, device family, manufacturer information, distributor information, etc. The database also contains a field for device unique identifier, but a value is only available for more recent reports and for devices that contain such information.

2.3.1.4 Patient Information

For each patient impacted by the event, the FDA also publishes releasable information in the MAUDE database. The included patient information contains treatment information, and outcome.

2.3.1.5 Narratives

A significant portion of the MedWatch forms are allocated to free-form narratives. These narratives describe the problem, specify relevant tests or laboratory data, other relevant history/preexisting conditions and manufacturer narrative.

2.3.2 Studies Using MAUDE

Several studies have been published in peer-reviewed literature utilizing the MAUDE database as the data source. For the most part, these studies have been narrow and focusing

on specific clinical specialties. We will briefly discuss a sampling of these studies in the sections below.

2.3.2.1 Studies on Medical Device Failure

(Hauser & Kallinen, 2004) conducted a search of the MAUDE database for product codes specific to all implantable cardioverter defibrillators (ICD) and the search term 'death'. The search result and the subsequent analysis found that 103 (69%) of a total of 150 reported ICD-related death events between 1996 and 2003 were associated with defective pulse generators or high-voltage leads in the ICDs (Hauser & Kallinen, 2004).

(DiBardino, McElhinney, Kaza, & Mayer, 2009) queried the MAUDE database for failures of a specific septal occluder device used to treat atrial septal defects and found reports of 223 failures in patients undergoing the device implantation procedure between 2002 and 2007. Of the adverse events, 17 (7.6%) were deaths and 152 (68.2%) needed a surgical rescue operation. The study also found that there was no significant difference in overall mortality between surgical and device closure, but mortality rates per adverse event were significantly higher in device closure (7.6%) compared to surgical closure (1.2%). The need for re-operation was also significantly higher (68%) per adverse event in device closure compared to surgical closure (3.6%) (DiBardino et al., 2009).

(Cope, Samuels-Reid, & Morrison, 2012) analyzed adverse events recorded in the MAUDE database involving insulin infusion pumps among pediatric population. The analysis showed that of 21,769 reports of insulin pump adverse events between 1996 and 2009, 1,774 (8%) were for children ages 1-12 years. More than half of these events had serious patient outcomes, including hospitalizations and 5 deaths (Cope et al., 2012).

(Manoucheri, Fuchs-Weizman, Cohen, Wang, & Einarsson, 2014) searched the MAUDE database for adverse events related to the use of a specific robot in gynecologic surgery. They found that between 2006 and 2012, there were a total of 280 cases of adverse events in the MAUDE database involving the use of the robotic device. 73 (26%) of the events resulted in injury and 24 (8.5%) resulted in death (Manoucheri et al., 2014).

(Andreoli, Lewandowski, Vogelzang, & Ryu, 2014) reviewed the MAUDE database to compare the complication rates associated with permanent and retrievable inferior vena cava filters (IVCF), a type of implantable medical device used to prevent pulmonary embolism. They found that between 2009 and 2012, there were 1,606 reported adverse events. 1,394 (86.8%) involved a retrievable IVCF, whereas 212 (13.2%) involved a permanent IVCF. While the true prevalence of IVCFs in the population is unknown, the study concluded that complications occur with significantly higher frequency with retrievable IVCFs compared to their permanent counterparts (Andreoli et al., 2014).

(Naumann & Brown, 2015) evaluated adverse events associated with electromechanical morcellation recorded in the MAUDE database. They found that between 2004 and 2014, there were 215 adverse events reported, of which 137 (64%) were due to device failures. 102 (74%) of the device failures resulted in conversion of the surgical procedure from minimally invasive to an open procedure. Nine of the adverse events resulted in death, mostly (8 of them) due to morcellation of unsuspected cancers, spreading the malignancy. The study also concluded that the reporting methodology that feeds data to the MAUDE database was suboptimal in capturing essential patient outcomes data (Naumann & Brown, 2015).

(Hebballi et al., 2015) reviewed 28,046 reports of adverse events in the MAUDE database associated with dental devices from 1996 to 2011. They found that 17,261 (61.5%) reported injuries, and 7,777 (28%) reported dental device malfunctions. 66 (0.2%) of the adverse events resulted in death. This study also highlighted that while the MAUDE database was the only available source of the information and held a potential to collect and share knowledge, it suffered from several limitations. These included insufficient information to determine the causes or factors contributing to an adverse event; sparse nature of the data contents; and potential reporting bias (e.g. devices that are expensive tend to be reported more because they are returned to the manufacturer for replacement more often and manufactures are *required* to report device failures) (Hebballi et al., 2015).

(Chen & Holsinger, 2016) queried the MAUDE database for morbidity and mortality associated with robotic head and neck surgery between 2009 and 2015. They found that 14 deaths and 11 injuries were associated with the use of the surgical robot. 8 of the deaths were between 2009 and 2011, which translated to a mortality rate of 0.3%, and consistent with a separate study. However the study also cited that the MAUDE database may not be capturing the normally expected proportions of surgical adverse events (Chen & Holsinger, 2016).

(Everett et al., 2016) extracted adverse event reports of stent fractures from the MAUDE database for years 2006 to 2011. They found 28 reports of bare metal stent and 481 reports of drug eluting stent (DES) fractures, which suggested a preference in use or reporting of DES fracture events. They also found that the clinical fracture reports were associated with the length of the stent and the use of multiple overlapping stents (Everett et al., 2016).

(Bielefeldt, 2017) searched the MAUDE database for adverse events between 2001 and 2015 (although relevant data was only available after 2008) involving gastric electrical stimulators (GES) devices used to treat patients with gastroparesis. The study of the search results found that there were 1,587 adverse events related to the GES, where a significant portion (35.7%) of the reported adverse events prompted surgical correction. The study concluded that there was a high likelihood of adverse events leading to secondary surgeries in patients undergoing GES and that the physicians needed to carefully weigh the risks of this intervention when counseling patients (Bielefeldt, 2017).

(Connor et al., 2017) studied the MAUDE database for adverse events involving radiation oncology devices (ROD) from 1991 to 2015. They found 4,234 adverse event reports involving RODs. The events were reported in external beam therapy (50.8%), brachytherapy – insertion of radioactive implants into the cancer tissue (24.9%) and treatment planning systems (21.6%). The major problems contributing to these adverse events were software (30.4%), mechanical (20.9%), and user error (20.4%). They also found that RODs experienced adverse events sooner after their manufacture or market approval compared with other devices (Connor et al., 2017).

2.3.2.2 Studies on Medical Device Computing Technology Failure

(Magrabi et al., 2011, 2012) searched and analyzed the MAUDE data between 2008 and 2010 for patient safety problems associated with healthcare information technology (HIT). They found 678 distinct reports of 436 different events. Of these, 46 (11%) were associated with patient harm. The harm included medication problems, clinical process problem, exposure to radiation and surgery problems. Another key finding of this study was

that only 1,100 (0.1%) of 899,768 reports of the events in MAUDE were related to HIT, suggesting a significant underreporting. (Magrabi et al., 2011, 2012).

(Magrabi et al., 2012) also classified the HIT adverse events into a set of 36 categories across "human", "machine", and "human-computer" spaces. The 436 different events entailed 712 problems. Most of these problems (682 or 96%) were machine-related, and 30 (4%) related to the human-computer interface.

(Alemzadeh et al., 2013) queried the MAUDE database looking for safety critical computer failures in medical devices. They found 75,267 reports of computer-related adverse events, including deaths. They also observed that there were some inconsistencies between the MAUDE database and the FDA's recalls database (Alemzadeh et al., 2013).

2.4 MACHINE LEARNING AND TEXT CLASSIFICATION

Artificial intelligence (AI) and machine learning have been the topics of immense interest in both academia and the industry in recent years, aided mainly by highly scalable and relatively inexpensive computing technologies. From personal assistants on mobile phones to autonomous vehicles, AI-based techniques have been successfully applied in solving problems that traditionally, required human reasoning – often extensively. Machine learning is one of the central elements of artificial intelligence – one that enables computers to perform actions based on a process of knowledge acquisition and refinement, rather than based on explicit programming. In the sections below, we will briefly survey some of the current trends in machine learning, with focus on its application in text classification – an area of our interest in this research.

2.4.1 Machine Learning

The modern history of machine learning dates back to the concept of the Turing Test (Turing, 1950), which established a measure for "intelligence" in computers still very relevant today: A computer passes the Turing Test if its response in a conversation cannot be distinguished by a human evaluator as having come from the computer or from a human participant. The first machine learning program is said to be a game of checkers developed at IBM by Arthur Samuel in which the computer incorporated 'learning' and refined its strategy with every game played (Samuel, 1959). Over the years, machine learning has established itself as a prominent research discipline in computer science and a number of applications have emerged across several industrial domains that rely significantly on machine learning-based techniques (Narula, 2017).

From an engineering perspective, *learning* can be defined as *changes in the system that are adaptive in the sense that they enable system to do the same task or tasks drawn from the same population more efficiently and more effectively next time* (Simon, 1983). This definition of learning as an adaptive refinement process is reflected in the following widely cited and formalized definition of machine learning:

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E (Mitchell, 1997).*

A machine learning task is generally approached from one of two broad learning paradigms: Supervised and Unsupervised. The decision to select the learning paradigm is based on the type of the data at hand and/or the end goal of the task. We will briefly discuss each of these paradigms and some of the most common techniques in the sections below.

2.4.1.1 Supervised Learning

Supervised learning is an approach of machine learning in which the computer is provided with a set of 'known' samples (called *labeled* data) and asked to make a prediction on a piece of unknown (*unlabeled*) data based on its knowledge of the patterns inherent in the known samples and their relationships with those in the unknown data vis-à-vis a predefined set of dimensions of interest, known as *features*. While the objective of a supervised learning activity is prediction (deterministic or probabilistic), the outcome may be expressed in terms of a discrete or continuous value depending on the functional goal of the activity typically either *classification* or *regression*.

In the classification problem, the computer assigns a *label* (also known as *class*) to an unknown data item based its match against the labeled dataset that it was trained on. So, the specific task is to really *classify* each unknown data item into one or more 'buckets', each representing a label. The classification may be *binary* (unknown data item belongs to one of two possible classes), *multi-class* (unknown data item belongs to one of three or more possible classes) or *multi-label* (unknown data item belongs to one or more possible classes). Two common techniques used in classification are *logistic regression* (Ng & Jordan, 2002) and *support vector machines* (Cortes & Vapnik, 1995). We will discuss both techniques at a greater detail in a later section in this chapter.

In the regression problem, the objective of the machine learning exercise is to yield a continuous value as the prediction for a given unknown input data item. Typically, *ordinary least square* (aka linear regression) technique is used in this method to predict the outcome. The computer, in this technique 'learns' by minimizing the *loss*, a measure of inaccuracy in its

predictions. We will discuss a popular method of minimizing the loss function known as *stochastic gradient descent* (Bottou, 2010) later in this chapter.

Parametric-techniques such as those discussed above assume that there is a relationship between a given input data item and the outcome. These techniques also assume that the relationship can be modeled by the computer. This assumption may not valid in many cases or too costly to realize. For such scenarios, non-parametric techniques such as *k-nearest neighbors* (prediction based on the labels of the nearest *k* data points) and *decision trees* (prediction based on splitting rules that minimize *entropy*, a measure of chaos) can be utilized that can infer outcomes based on the labeled data.

Our research utilizes the supervised learning paradigm. Specific techniques and parameters used will be discussed in later sections.

2.4.1.2 Unsupervised Learning

Arguably the most important asset (and bottleneck) in a supervised machine learning task is labeled data, which can be rare and expensive to generate. Unsupervised learning is an approach to apply machine learning on datasets that do not have predefined labeled samples. The objective of an unsupervised machine learning activity is also significantly different than that of a supervised learning activity. Whereas in supervised learning the objective is to predict a value of the dependent variable for a given unknown data item, unsupervised learning is used mainly to understand the structure of the data. Typical functional goals of an unsupervised learning exercise include *clustering* and *dimensionality reduction*.

Clustering is a method of grouping data points by their similarity. Two popular techniques of clustering include *k-means clustering* (define *k* clusters and assign each data point

to one based on its similarity) and *hierarchical clustering* (establishing a hierarchical relationship among clusters the data is grouped into).

Dimensionality reduction is a feature extraction activity that transforms the dimensional space in the input dataset to a smaller size by identifying and extracting significant features. This can help reduce cost and complexity of machine learning on high-dimension datasets. *Principal component analysis* (linear transformation of the data to a lower dimensional space) is a common technique used in dimensionality reduction methods.

In recent years, a concept known as *deep learning* is also gaining significant interest in the research community. Deep learning is a method (supervised or unsupervised) of machine learning that utilizes artificial neural networks with backpropagation on a massive scale.

Much like statistics, machine learning is not a perfect science. Thus, it is not suitable for problems that require 100% accuracy at all times. However, its importance in helping us make sense of the universe cannot be overstated. Some have predicted that machine learning, however inefficient, may still be more efficient than programming (Simon, 1983). The increasing adoption of machine learning in industrial and consumer applications we are experiencing today proves its utility and point us to a potentially new era in computing.

2.4.2 Text Classification using Machine Learning

Information retrieval from textual documents, generally known as text data mining, has a long history (J. Jiang, 2012; Sebastiani, 2002). Until the 1980s, statically-defined, rule-based *knowledge engineering* was the main approach used in the industry for typical text classification or categorization tasks. However, advancements in machine learning techniques and abundance of inexpensive computing power has made text mining one of the

most common applications in machine learning in recent years. We will briefly discuss some of the common machine learning use cases in text mining below.

2.4.2.1 Named Entity Recognition

Named entity recognition (NER), one of the most fundamental tasks in information retrieval, is a process of identifying instances of real-world objects ('*named entities*') from a block of text and classifying them into a predefined set of types such as person, organization or location (J. Jiang, 2012). Machine learning NER techniques include rule-based methods, where the machine automatically learns the rules, or statistical learning methods such as *Markov models*, a probabilistic approach of establishing classification based on just one (*first order Markovian*) or a few prior classifications.

2.4.2.2 Text Classification

In formal terms, text classification can be defined as the task of assigning a Boolean value to each pair $(d_j, c_i) \in D \times C$, where D is a domain of documents and $C = \{c_1, \ldots, c_{|C|}\}$ is a set of predefined categories (Sebastiani, 2002). In simpler terms, it is a process of assigning one or more pre-defined domain-specific labels (or *classes*) to a block of text. Typically, the classification is based on the significance of the presence of *features* that are generated from pre-labeled training data.

A large number of machine learning models are applicable to a text classification task. These include logistic regression, support vector machine, decision trees and probabilistic methods such as naïve Bayes. The machine learning portion of our research is a supervised natural language text classification problem. We will discuss the theoretical underpinnings of the models used in our research at a greater detail later in this chapter.

2.4.2.3 Relationship Extraction

Relationship extraction is the task of detecting and characterizing prespecified types of semantic relations between entities in text (Cohen & Hersh, 2005; J. Jiang, 2012). A wide range of techniques are available for relationship extraction. These include defining relationship templates with patterns; statistical methods; and a technique we utilize in our research known as *snowballing*, where a small seed of sample data is used to iteratively find and expand new relations in a large corpus. We will discuss our approach in a greater detail in the Chapter III.

2.4.3 Models and Techniques

A number of models and techniques have been defined over the years for various forms of machine learning tasks. In the sections below, we will analyze the theoretical basis for some of the models and techniques we use in this research.

2.4.3.1 Naïve Bayes

Naïve Bayes is a probabilistic classification scheme built on the Bayes' Theorem, which describes the probability of an event based on known prior probabilities or *priors* – making it a suitable candidate for supervised (i.e. prior labeled data exists to compute the priors) machine learning. We utilize this model in our experiments considering its relative strength in text classification problems (Lewis, 1998).

At the fundamental level, Bayes' Theorem is defined as:

$$P(B|A) = \frac{P(B)P(A|B)}{P(A)} \quad (1)$$

Applying this conditional probability scheme to a classification problem, we get:

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)} \quad (2)$$

Where:

c is the classification, or *class*, (in our case binary, *positive* or *negative*); and d is the document for which the classification is sought. Now, recall that we represent every document d in the form of a feature vector ($x \in X$), where the vector contains a set of numbers each representing the normalized count of the occurrence of a feature in the document. For example, Table 8 shows a simplistic (and fictional) visualization of a set of vectors across a set of training documents (each *row* is a feature vector representing the document in our training set):

Training Document	**Feature x_1 ('robot')**	**Feature x_2 ('leak')**	**Feature x_3 ('keyboard')**	**Feature x_n ('motherboard')**	**Class (y)**
$\boldsymbol{d_1}$	$x_1 = 1$	$x_2 = 0$	$x_3 = 0$	$x_n = 1$	Pos
$\boldsymbol{d_2}$	$x_1 = 2$	$x_2 = 2$	$x_3 = 0$	$x_n = 0$	Neg
$\boldsymbol{d_3}$	$x_1 = 1$	$x_2 = 1$	$x_3 = 0$	$x_n = 1$	Pos
$\boldsymbol{d_j}$	$x_1 = 0$	$x_2 = 0$	$x_3 = 1$	$x_n = 1$	Pos

Table 8: Feature Vector Examples

With this, what we know is that $P(d)$ is really a joint probability of its feature vector:

$$P(c|x_1, x_2, x_3, \ldots x_n) = \frac{P(c)P(x_1, x_2, x_3, \ldots x_n|c)}{P(x_1, x_2, x_3, \ldots x_n)} \quad (3)$$

Using the multiplication rule of probabilities:

$$P(c|x_1, x_2, x_3, \dots x_n) = \frac{P(x_1, x_2, x_3, \dots x_n, c)}{P(x_1, x_2, x_3, \dots x_n)} \quad (4)$$

An issue in this strategy is that computing the joint probability $P(x_1, x_2, x_3, \dots x_n, c)$ can be expensive if the number of features is very high. It also requires that all possibilities are reflected in the data that the model can see at the time of training:

$$\begin{aligned} & P(x_1, x_2, x_3, \dots x_n, c) \\ &= P(x_1|x_2, x_3, \dots x_n, c)P(x_2, x_3, \dots x_n, c) \\ &= P(x_1|x_2, x_3, \dots x_n, c)P(x_2|x_3, \dots x_n, c)\, P(x_3, \dots x_n, c) \\ &= \dots \\ &= P(x_1|x_2, x_3, \dots x_n, c)P(x_2|x_3, \dots x_n, c)P(x_3, \dots x_n, c) \dots P(x_{n-1}|x_n, c)P(c) \end{aligned} \quad (5)$$

To address these issues, an important assumption made in this model (hence *naïve* in the name) is that each feature is independent from another in its ability to influence the outcome. This assumption can be expressed as:

$$P(x_i|x_1, x_2, x_3, \dots, x_{i-1}, x_{i+1}, \dots x_n, c) = P(x_i|c) \quad (6)$$

This assumption now allows us to contain the growth in the number of parameters and computational complexity shown in (5), as all joint probabilities of the features can now be replaced with the probability of the feature being evaluated given the class under consideration:

$$P(c|x_1, x_2, x_3, \dots x_n) = \left(\prod_{i=1}^{n} P(x_i|c)\right) \frac{P(c)}{P(x_1, x_2, x_3, \dots x_n)} \quad (7)$$

With this, we now calculate the probability of a class given the document feature vector by evaluating for all classes, in our case $c \in C = \{positive, negative\}$.

One additional optimization that is performed in this model is the elimination of the denominator because $P(x_1, x_2, x_3, \dots x_n)$ is the same no matter what class $c \in C$ the equation is being evaluated for. The consequence of this elimination, however, is that the outcome is no longer a probability estimate, but instead a value that would be directly proportional to it:

$$P(c|x_1, x_2, x_3, \dots x_n) \propto \left(\prod_{i=1}^{n} P(x_i|c)\right) P(c) \tag{8}$$

With this, the classification function in this model is really an $argmax$ function on the argument $c \in C$ that maximizes the value of the equation in (8):

$$\hat{c} = \underset{c \in C}{argmax} \left(\prod_{i=1}^{n} P(x_i|c)\right) P(c) \tag{9}$$

2.4.3.2 Linear Regression

Linear regression models estimate the scalar value of a dependent variable for a set of feature vectors (independent variables) using a linear function. For example, in simple linear regression, the predictor function has the following form for any i-th feature vector x:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i \tag{10}$$

Where β_0 is the constant (intercept), β_1 the coefficient of x (slope), and ε is the statistical error. The goal of a linear regression activity is to find a linear function that best fits the observations in the training data. This is done by estimating and adjusting the parameters β_0 and $\beta_1$to minimize ε using techniques such as the *ordinary least squares* (OLS):

$$\min_{\beta_0,\beta_1} f(\beta_0,\beta_1) = \sum_{i=1}^{n} \hat{\varepsilon}_i^{\,2} \tag{11}$$

Where $\hat{\varepsilon}$ is the residual (or distance) between the actual value (y) and the value of the dependent variable predicted by the model under training. Given $\hat{\varepsilon}$ is estimated using (10), we can express (11) as:

$$\min_{\beta_0,\beta_1} f(\beta_0,\beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2 \tag{12}$$

In addition to OLS, there are other techniques also available to best estimate the parameters β_0 and β_1. These include *Ridge* (Hoerl & Kennard, 1970) and least absolute shrinkage and selection operator or *Lasso* (Tibshirani, 1996) techniques. We will not discuss those here, as we do not utilize them in our research.

2.4.3.3 Logistic Regression

A general assumption in the linear regression model is that the outcome values are continuous and that they hold a linear relationship with feature vectors, or predictors. Often times, however, the outcome is not continuous, but rather dichotomous (e.g. true/false or positive/negative) or categorical. When a linear regression technique is applied on this type of data, the predictor function can yield values outside the range of a probability estimate (0 - 1), rendering the approach of little utility on this type of data.

Despite its name, logistic regression is a *classification* (note, not regression) method that is best suited for classifying feature vectors into dichotomous outcomes. It solves the problem of out-of-range probability estimates of linear regression by effectively establishing a floor (near 0) and a ceiling (near 1) for all outcomes utilizing the *logit* transformation which defines a natural log of the odds ratio:

$$logit(p) = log_e\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x \tag{13}$$

Where e is the Euler's number $\cong 2.71828$. With this transformation, we predict the natural log of the odds ratio of the event being 'success' or 1. It is also possible to express this in terms of probability (p) with some additional transformation. Recall (13):

$$log_e\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x$$

Raising both sides by e:

$$e^{log_e\left(\frac{p}{1-p}\right)} = e^{(\beta_0 + \beta_1 x)} \tag{14}$$

Given natural logarithm log_e is the inverse of e: $e^{\log_e(x)} = x$:

$$\frac{p}{1-p} = e^{(\beta_0 + \beta_1 x)} \tag{15}$$

Multiplying both sides by $1 - p$:

$$p = e^{(\beta_0 + \beta_1 x)} - (p \times e^{(\beta_0 + \beta_1 x)}) \tag{16}$$

Solving for $e^{(\beta_0 + \beta_1 x)}$:

$$e^{(\beta_0 + \beta_1 x)} = p + (p \times e^{(\beta_0 + \beta_1 x)})$$
$$e^{(\beta_0 + \beta_1 x)} = p(1 + e^{(\beta_0 + \beta_1 x)}) \quad (17)$$

Solving for p:

$$p = \frac{e^{(\beta_0 + \beta_1 x)}}{1 + e^{(\beta_0 + \beta_1 x)}} \quad (18)$$

The outcome of this predictor is a probabilistic estimate on feature vector x that the dependent variable equals "success" (or 1 on the logistic curve).

2.4.3.4 Support Vector Machine

When data forms a linearly separable pattern into dichotomous classes with respect to some features, the support vector machine (SVM) algorithm (Cortes & Vapnik, 1995) helps establish an optimal hyperplane that separates the data. In a binary classification problem, such as ours, this technique is very useful in maximizing the confidence in the model's predictions.

The objective of a linear classification model is to establish a line (or a *hyperplane* in a multi-dimensional space) that separates the classes of interest in the data. In simplistic terms, SVM is a technique to draw this hyperplane in an optimal way. Consider the plot in Figure 4 on some two-dimensional plane where data is linearly separable into *positive* (+) and *negative* (-) classes. Which of the lines (or some other) in the plot should the predictor use to tell whether a new observation is positive or negative?

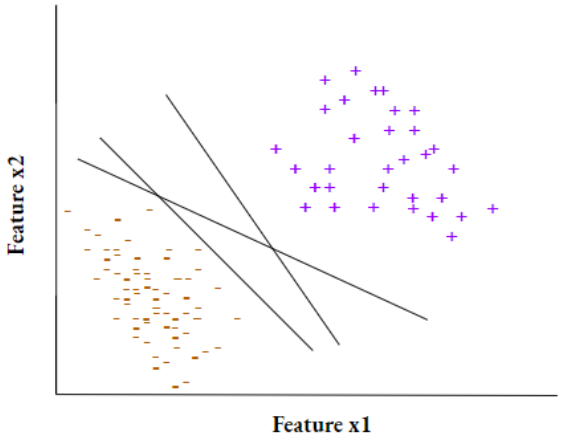


Figure 4: Linearly Separable Data

There are infinite possibilities to draw a hyperplane between two linearly separable spaces. Support vector machine computes an "optimal" hyperplane by determining Support Vectors (points closest from the hyperplane on either side) and maximizing the margin between them. For example, for the illustration in Figure 4, a corresponding illustration with an SVM-optimized hyperplane may look like the plot in Figure 5:

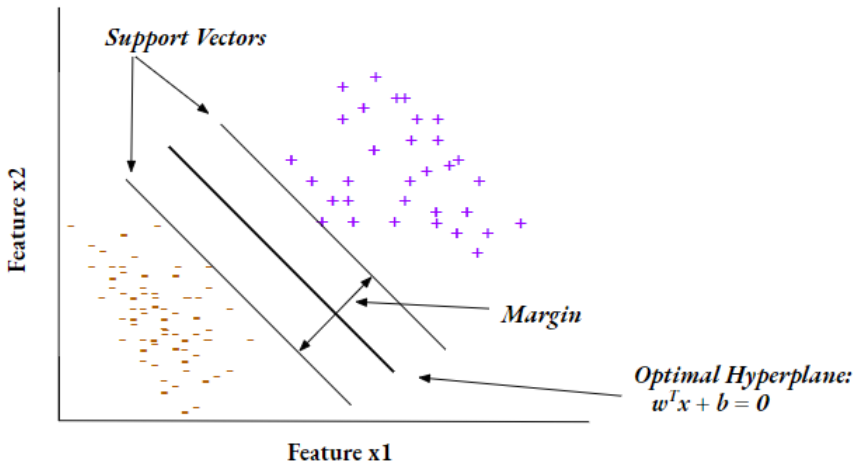


Figure 5: Support Vector Machine Hyperplane

Let us briefly analyze the mechanism for deriving the optimal hyperplane. Note that on a 2-dimensional plane, a line is defined as:

$$y = mx + b$$

or:

$$mx - y + b = 0 \tag{19}$$

Let us re-label x and y as x_1 and x_2, respectively (this will allow for a more convenient mapping to an n-dimensional feature vector, which by convention uses the $x_1 \ldots x_n$ notation as shown in Table 8):

$$mx_1 - x_2 + b = 0 \tag{20}$$

In two dimensions, this equation is satisfied by the dot product of two specific vectors that contain these components:

One vector, call it $x = (x_1, x_2)$; and another which we will call $w = (m, -1)$. We can prove this by computing the dot product $x \cdot w$:

$$\begin{aligned} x \cdot w &= x_1 m + (-1 \times x_2) \\ &= x_1 m - x_2 \end{aligned} \tag{21}$$

With this, we can now restate equation of the hyperplane as:

$$x \cdot w + b = 0 \tag{22}$$

Given $x \cdot y = x^T y = y^T x$, we can also express this as:

$$w^T x + b = 0 \tag{23}$$

In SVM terms, x is the *input* vector of n dimensions with each component representing a feature; w a *weight* vector of the same dimension -- each component representing the weight of the corresponding feature, and b a scalar *bias*. The values for the weight vector and bias are determined through a process of optimization during the model's training. Assuming dichotomous class $y \in \{positive, negative\}$, the hyperplane is fit to accurately classify the training data with the following constraints:

$$w^T x + b < 0 \; for \; all \; x \; that \; belong \; to \; class \; negative$$
$$w^T x + b > 0 \; for \; all \; x \; that \; belong \; to \; class \; positive \tag{24}$$

We illustrate these constrains in Figure 6 below:

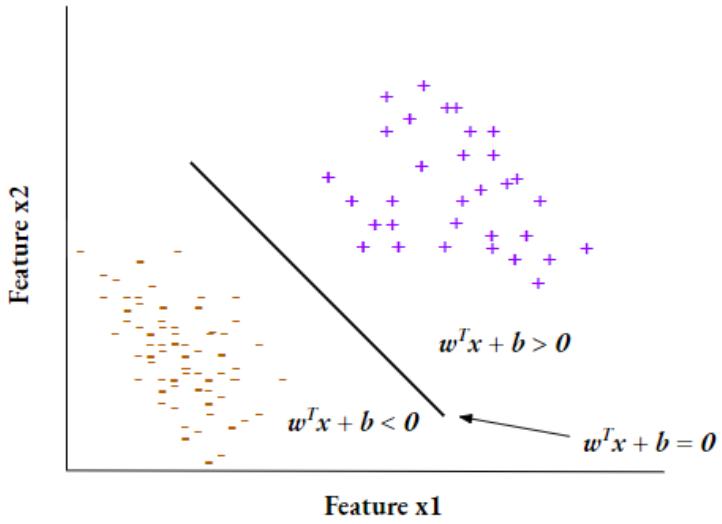


Figure 6: Constraints of the SVM Hyperplane

Note that the class labels are domain-specific. The only assumption SVM makes is that they are dichotomous. Their specific values (positive/negative; yes/no; computer-related/computer not related, etc.) are irrelevant to the model.

A key distinguishing feature of the SVM technique (compared, to other linear methods such as linear regression, which also essentially "draw" a line as a decision boundary) is that SVM establishes the optimal hyperplane (let us call it H_0) based on support vectors (see Figure 5) which are also hyperplanes.

Assuming a training set with a collection of vectors $x_i \in \boldsymbol{R^n}$ and the associated dichotomous classes $y_i \in \{-1, +1\}$ (descriptive class labels such as *negative, positive* must be mapped to the {-1 and +1} classes), we can define the hyperplane H_0 using (23) with the following constraints:

$$\begin{aligned} w^T x_i + b &\geq +1 \; for \; all \; x_i \; where \; y_i \; is + 1 \\ w^T x_i + b &\leq -1 \; for \; all \; x_i \; where \; y_i \; is - 1 \end{aligned} \tag{25}$$

The hyperplanes (H_0, H_1, H_2) are defined where the margin is maximized subject to constraints in (25). Figure 7 illustrates their relationship:

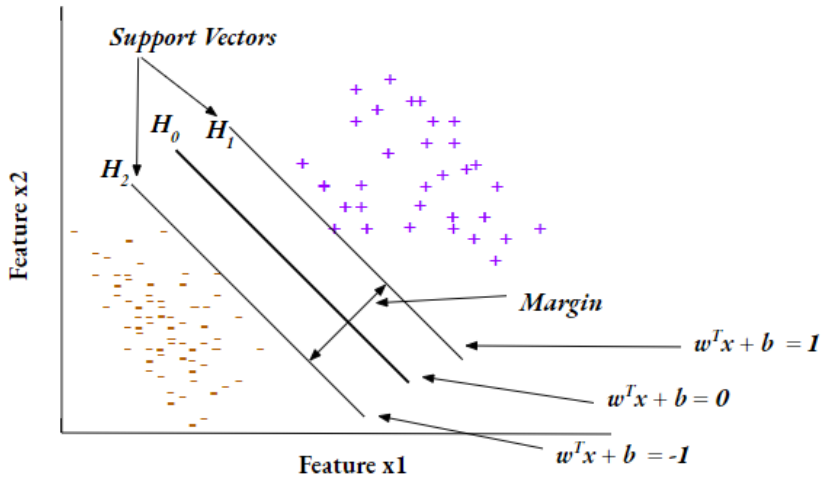


Figure 7: SVM Support Hyperplanes

Given $y_i \in \{-1, +1\}$, the constraints in (25) can be consolidated utilizing y_i:

$$y_i(w^T x_i + b) \geq 1 \tag{26}$$

The margin (i.e. distance) between the two supporting hyperplanes (H_1, H_2) can be represented as $\frac{2}{||w||}$ (Gunn, 1998). This is a quadratic programming problem, where we maximize $\frac{2}{||w||}$ subject to the constraints in (26). Conversely, given $||w||$ is a positive number and $\frac{2}{||w||}$ is a monotonically decreasing function, the maximization problem can be converted into a minimization problem:

$$\min_{w} \frac{1}{2}||w||^2 \; s.t. \;\; y_i(w^T x_i + b) \geq 1 \, , i = 1, \dots, n \tag{27}$$

Thus, the optimal support vector machine hyperplane ($w^T x + b = 0$) is one in which w and b are obtained from the minimization process (27).

2.4.3.5 Gradient Descent

All linear predictors contain some error, which, for a given feature vector, is the distance (*residual*) between the actual (y) and the predicted ($\hat{y}$) values:

$$error = y - \hat{y} \tag{28}$$

At the model level, for a given training set of (x_i, y_i), where x_i is an nth dimensional vector, and y_i its associated class, we can express the total error of the model as a sum of all squared errors in the training set:

$$total\ error = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{29}$$

We can distribute the total error (29) across the training set to obtain a normalized error factor for a new prediction. This, for example, can be achieved by computing the arithmetic mean of the squared errors:

$$mean\ error = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{30}$$

In the case of a linear predictor, $\hat{y}_i = mx_i + b$, we can represent the error of m and b as a *loss function*:

$$loss(m, b) = \frac{1}{n} \sum_{i=1}^{n} (y_i - (mx_i + b))^2 \tag{31}$$

Gradient descent (Rumelhart, Hinton, & Williams, 1986) is a technique to optimize the parameters (m, b) of the hypothesis function so that the error (or *loss*) is minimized. The technique involves iteratively reducing the loss and adjusting the parameters in every step of the iteration until the parameters can no longer be optimized. The optimization is performed by computing the gradient (derivative) of the loss function relative to parameter(s) being optimized. We will analyze this process below.

For a model in training, let us denote the $error$ of a particular feature vector i as a function of the parameters to be optimized:

$$error_i(m, b) = y_i - (mx_i + b) \tag{32}$$

We can now restate the loss function as:

$$loss(m, b) = \frac{1}{n}\sum_{i=1}^{n}(error_i(m, b))^2 \tag{33}$$

This dependency of the $loss$ function on the $error$ function can also be stated in an expanded form as:

$$loss(error_i(m, b)) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (mx_i + b))^2 \tag{34}$$

As stated earlier, the technique used in gradient descent to optimize the parameters is through the computation of the gradient of the loss function relative to the parameters being optimized. In the running example, we have two parameters: (m, b). So, we will obtain the

gradient for each separately by computing the partial derivative of the $loss$ function relative to m and b.

Recall the chain rule of the derivative of the composition of two functions $F(x) = f\big(g(x)\big)$:

$$F'(x) = f'\big(g(x)\big)g'(x) \tag{35}$$

Using (34) and (35):

$$F'(m,b) = loss'(error_i(m,b))\ error'_i(m,b) \tag{36}$$

Let us first calculate the derivative $loss'(error_i(m,b))$ relative to m:

$$\frac{\partial}{\partial m}\ loss(error_i(m,b)) \tag{37}$$

Using (34):

$$\frac{\partial}{\partial m}\frac{1}{n}\sum_{i=1}^{n}(y_i - (mx_i + b))^2 \tag{38}$$

Using the power rule of derivatives, where for any $f(x) = mx^n$, derivative $f'(x) = mn \times x^{n-1}$:

$$= 2 \times\frac{1}{n}\sum_{i=1}^{n}(y_i - (mx_i + b))^{2-1} \tag{39}$$

$$= \frac{2}{n}\sum_{i=1}^{n}(y_i - (mx_i + b)) \quad (40)$$

This analysis shows that the partial derivative of function $loss(error_i(m, b))$ relative to m is (40). Now, let us calculate the derivative $error'_i(m, b)$ relative to m:

$$\frac{\partial}{\partial m}\, error(m, b) \quad (41)$$

Substituting the definition of function $error$:

$$\frac{\partial}{\partial m}\,(y_i - (mx_i + b)) \quad (42)$$

Given b and y are constants (in the context of a partial derivate relative to m), their derivative is 0:

$$= \frac{\partial}{\partial m}\,(0 - (x_i\, m + 0)) \quad (43)$$

$$= \frac{\partial}{\partial m}\,(-x_i\, m^1) \quad (44)$$

Note, the raising of m to the $1th$ power has no effect. It is only to make the expression obvious for the power rule of derivative, which states that for all $f(x) = x^n$, derivate $f'(x) = n \times x^{n-1}$:

$$\frac{\partial}{\partial m}\,(-x_i\, m^1) = (1 \times -x_i\, m^{1-1}) \quad (45)$$

$$= -x_i \quad (46)$$

This analysis shows that the partial derivative of function $error(m, b)$ relative to m is $-x_i$:

$$\frac{\partial}{\partial m}\, error(m, b) = -x_i \tag{47}$$

Now, using (40) and (47) and multiplying the derivatives $loss'(error_i(m, b))$ and $error'_i(m, b)$ per (36), the partial derivative of the $loss$ function relative to m is:

$$\frac{2}{n}\sum_{i=1}^{n} -x_i\,(y_i - (mx_i + b)) \tag{48}$$

If we follow the same process to obtain the partial derivative of the $loss$ function relative to b, we achieve:

$$\frac{2}{n}\sum_{i=1}^{n} -1\,(y_i - (mx_i + b)) \tag{49}$$

This (49) is because $loss'(error_i(m, b))$ relative to b is the same as $loss'(error_i(m, b))$ relative to m and the partial derivative $error'_i(m, b)$ relative to b works out to be -1 given mx and y are constant (so derivative = 0) and the power rule $f'(x) = n \times x^{n-1}$:

$$= \frac{\partial}{\partial b}\,(0 - (0 + b)) \tag{50}$$

$$= \frac{\partial}{\partial b} (-b^1) \tag{51}$$

$$= -1 \times b^{1-1} \tag{52}$$

With this analysis, we have shown how the gradience values can be calculated to optimize the parameters (m, b). Typically, a gradient descent-based learning technique adjusts the parameters based on the gradience values, and a *learning rate*. Learning rate can be a constant or an adaptive function that allows for larger adjustments (higher learning rate) early in the optimization process, but slows (lower learning rate) near *convergence*, point where the loss is minimum. Assuming a learning rate of α, we can update the optimal values for m and b as follows:

$$m := m - \alpha \frac{2}{n} \sum_{i=1}^{n} -x_i (y_i - (mx_i + b)) \tag{53}$$

$$b := b - \alpha \frac{2}{n} \sum_{i=1}^{n} -(y_i - (mx_i + b)) \tag{54}$$

One problem with this (53, 54) approach, known as *batch gradient descent*, is that the computation of the gradience value requires iterating through the entire training set every time the adjustment needs to be performed. This can be very expensive when the training set is large. One major improvement, known as *stochastic gradient descent* (Bottou, 2010) eliminates the iterative computation/summation of the partial derivatives, and instead uses the derivative of a *randomly* selected single training sample to approximate the gradience at every step until convergence.

We define a new loss function $stochastic\_loss$ that only looks at the sample at hand:

$$stochastic\_loss(m, b) = error(m, b)^2$$
$$= (y_i - (mx_i + b))^2 \quad (55)$$

This would mean that:

$$\frac{\partial}{\partial m}\, stochastic\_loss\big(error_i(m, b)\big) =$$
$$\frac{\partial}{\partial m}\, (y_i - (mx_i + b))^2 \quad (56)$$

Using the power rule of derivatives:

$$= 2(y_i - (mx_i + b)) \quad (57)$$

This means, using (36) and the subsequent analysis, the derivative $stochastic\_loss'(m, b)$ relative to each of m and b are:

$$\frac{\partial}{\partial m} stochastic\_loss(m, b) = -2x_i\,(y_i - (mx_i + b)) \quad (58)$$

$$\frac{\partial}{\partial b} stochastic\_loss(m, b) = -2(y_i - (mx_i + b)) \quad (59)$$

Thus, with the learning rate of a, the update rules for the *stochastic* gradient descent for m and b would be:

$$m := m - (-2ax_i\,(y_i - (mx_i + b))) \quad (60)$$

$$b := b - (-2a(y_i - (mx_i + b))) \tag{61}$$

We discuss our application of these models in sections 3.2.3 and 3.2.5 in Chapter III. It should be noted that the application of these models in our research is through a set of machine learning libraries (Pedregosa et al., 2011) implemented in the scientific computation modules of the Python programming language. The formal analyses we performed were for our own understanding and validation of the theoretical underpinnings behind these models. We hope that our documentation of the analyses in this and the preceding sections will help the Reader with learning goals, as we did not find similarly detailed derivations described in the existing literature.

CHAPTER III: METHODS

3.1 INTRODUCTION

This research is comprised of two major components:

1. Identification of medical device events related to computing technology;

2. Analysis of medical device failures related to computing technology.

For the identification component of the research, we implemented a supervised machine learning method. Our goal was to categorize each report of medical device event in the Manufacturer and User Facility Device Experience (MAUDE) dataset published by the U.S. Food and Drug Administration (FDA) into either *positive* or *negative* class, representing computing technology related, and not related events, respectively. More specifically, we designed a multi-model supervised text classification technique that classified over 11 million natural language narrative records in the MAUDE dataset.

The design of the machine learning experiment involved model selection, training data generation, trained model generation, classification and verification steps. To generate the training data necessary to train computer models, we designed and implemented a seed-based snowballing scheme. Then we used the generated training data to train classifiers based on three different models (Naïve Bayes, Logistic Regression, Support Vector

Machine). The trained models were then used to classify each of the over 11 million narrative records into a dichotomous set of classes. The result of the classification was verified through automated checks and manual reviews, as well as through various statistical measures.

For the analysis component of this research, we joined the results of the classification task with other files also in the MAUDE dataset that contained additional information on each medical device failure event. We imported all data into an analysis platform and perform targeted queries and analytics around our research questions.

We describe the methods used in our research in greater detail in the sections below.

3.2 IDENTIFICATION OF RECORDS OF INTEREST

As discussed in Section 1.2, the problem code assignment in MAUDE database appeared to be masking medical device failures related to computing technology. In this research, we designed an alternative, machine learning-based method of identifying failures related to computing technology.

3.2.1 Goal

The ultimate objective of the machine learning activity in our research was to categorize each report of medical device event in the MAUDE dataset into one of two classes:

1. ***Positive:*** Medical device failure related to computing technology
2. ***Negative:*** Medical device failure not related to computing technology

3.2.2 Data Source

Data for this research was obtained from the FDA's MAUDE database portal on the World Wide Web as described in Section 1.2.1.1. For the machine learning experiments, we extracted the archive (zip) files containing the narrative records for each of the study years (2007 – 2016) into a text file. Table 9 lists the file names and the number data records contained:

Source File Name	**Extracted File Name**	**Description**	**Number of Data Records**
foitext2007.zip	foitext2007.txt	Text narrative records for year 2007.	232,626
foitext2008.zip	foitext2008.txt	Text narrative records for year 2008.	264,971
foitext2009.zip	foitext2009.txt	Text narrative records for year 2009.	388,041
foitext2010.zip	foitext2010.txt	Text narrative records for year 2010.	697,472
foitext2011.zip	foitext2011.txt	Text narrative records for year 2011.	972,480
foitext2012.zip	foitext2012.txt	Text narrative records for year 2012.	1,251,520
foitext2013.zip	foitext2013.txt	Text narrative records for year 2013.	1,536,462

Source File Name	Extracted File Name	Description	Number of Data Records
foitext2014.zip	foitext2014.txt	Text narrative records for year 2014.	1,965,058
foitext2015.zip	foitext2015.txt	Text narrative records for year 2015.	2,274,087
foitext2016.zip	foitext2016.txt	Text narrative records for year 2016.	2,106,765
		Total Records:	**11,689,482**

Table 9: Number of Narrative Records in MAUDE

Each of these files also included a header record (not included in the counts in Table 9) describing the data fields. We list these fields in the section below.

3.2.2.1 Record Format

Each narrative record in MAUDE is a pipe (|) character-delimited sequence of values for the following fields:

Field Name	Description
MDR_REPORT_KEY	Identifier for the failure report. This field joins the narrative with the event information in other files.
MDR_TEXT_KEY	Identifier for the narrative text.

Field Name	Description
TEXT_TYPE_CODE	The MedWatch forms (See Appendix A and B) used to submit reports of device failure contains multiple sections where text narrative can be entered. This single character field identifies the section on the MedWatch form where the narrative was entered. 'D' = Text entered in section B5 of the form 'E' = Text entered in section H3 of the form 'N' = Text entered in section H10 of the form
PATIENT_SEQUENCE_NUMBER	Patient sequence number. This is not a patient identifier. Instead, it is a number (starting with 1) assigned to each patient, whose treatment/outcome information is included in the submission.
DATE_REPORT	Date of the report.
FOI_TEXT	The text narrative describing the event.

Table 10: Format of Narrative Records in MAUDE

Our machine learning experiments utilized the contents in FOI_TEXT, MDR_REPORT_KEY and MDR_TEXT_KEY fields in each record. The FOI_TEXT field was used for text classification and MDR_REPORT_KEY field was used to map the record to information of interest in other files. MDR_TEXT_KEY was used to explore the 1:M

relationship between an event report and its associated narrative records. We ignored the other fields.

3.2.3 Model Selection

Selection of models for a machine learning-based text classification task is not a precise science, and rather an experimental process. We considered a variety of models for our machine learning task, and shortlisted the following models as candidates based on their applicability and popularity:

3.2.3.1 Naïve Bayesian Classifier

Naive Bayesian classifier is a probabilistic model of assigning classification to a document based on features. The classifier calculates the probability of each class $c \in C$ given the feature vector and assigns classification with the highest probability. Formally, this classifier can be expressed as an $argmax$ function on the argument $c \in C$ that maximizes the probability of a classification given a feature vector (See our detailed analysis in Section 2.4.3.1):

$$\hat{c} = \underset{c \in C}{argmax} \left(\prod_{i=1}^{n} P(x_i|c) \right) P(c) \tag{62}$$

3.2.3.2 Logistic Regression Classifier

Logistic regression-based classifiers are designed to classify feature vectors into dichotomous classes through logit transformation, which effectively solves the problem of potential out-of-range probability values when linear regression-based estimation is applied on binary classes. Formally, the classifier's estimated probability of a feature vector equaling

the 'success' class (e.g. *Positive* in our case) can be expressed as (See our detailed analysis in Section 2.4.3.3):

$$p = \frac{e^{(\beta_0 + \beta_1 x)}}{1 + e^{(\beta_0 + \beta_1 x)}} \tag{63}$$

3.2.3.3 Support Vector Machine

Support vector machine is a linear classification technique that maximizes the margin between the hyperplanes that separate the classes in data. Classification is based on a feature vector's position relative to the optimal hyperplane. The optimal hyperplane is established by maximizing the margin between two support vectors each representing the nearest data point between the two classes. On a two-dimensional plane, the optimal hyperplane can be expressed as:

$$w^T x + b = 0 \tag{64}$$

In which, w and b are obtained from the following minimization process (See our detailed analysis in Section 2.4.3.4):

$$\min_{w} \frac{1}{2} ||w||^2 \; s.t. \;\; y_i(w^T x_i + b) \geq 1 \, , i = 1, \dots, n \tag{65}$$

3.2.3.4 Stochastic Gradient Descent

Stochastic Gradient Descent is a machine learning technique of minimizing errors in the model's predictions. The process involves iterating through the training data and progressively minimizing the loss function until convergence. For a linear predictor $y =$

$mx + b$, what this means is that the parameters (m, b) are iteratively optimized such that the loss is minimized. The specific *stochastic* gradient descent algorithm to update these parameters can be expressed as (See our detailed analysis in Section 2.4.3.5):

$$m := m - (-2ax_i\,(y_i - (mx_i + b))) \tag{66}$$

$$b := b - (-2a(y_i - (mx_i + b))) \tag{67}$$

Our application of the support vector machine model in this research utilizes stochastic gradient descent technique for learning.

3.2.3.5 Model Ensemble

We recognized that choosing the best model for the classification task was going to be an empirical process (Sebastiani, 2002). Our initial approach was to test the performance of each of the models, but no clearly differentiating classifier emerged. We then decided to implement an increasingly popular ensemble technique (Domingos, 2012) using all three models in a voting scheme, where an overall positive classification required all the three models to classify a record as positive.

3.2.4 Training Data Generation

A supervised machine learning task requires training data. In a classification problem, the training data is *labeled*, or pre-classified, which the computer then uses to compute the model parameters for prediction on previously unseen data. In our case, however, no labeled data existed.

We implemented a novel technique that combined elements of a snowball system of extracting relations (Agichtein & Gravano, 2000) with a quality-controlled auto-labeling approach, which we designed ourselves.

3.2.4.1 Goal

The objective of the *Training Data Generation* activity was to produce sufficient number of labeled samples from the 11,689,482 records in the corpus. Specifically, we set the following goals:

1. **Size:** Generate at least 116,895 (1% of the corpus) labeled records divided roughly evenly between positive and negative samples;
2. **Distribution:** Ensure sampling from all 10 files in the corpus.

3.2.4.2 Quality-Controlled Auto-Labeling

We considered various different approaches of generating the training data. Initially, we followed a simple, manual method where we sequentially examined each record and placed it in one of two files representing the classes (positive, negative). However, there were two problems with this approach:

1. It took roughly 1 minute to manually classify each record. At this rate, it would take nearly 2,000 hours to generate the training data; and
2. Due to the sparsity of positive records in the corpus, negative records would build up quickly while only a much smaller set of positive records could be identified.

We quickly realized that the manual approach would not be feasible. We also considered crowdsourcing the labeling work; but the effectiveness, quality, cost and logistical

complexity of such an approach could not be confirmed. We then evaluated alternative methods, traditionally used in information extraction and text data mining. These included, *snowballing* (Agichtein & Gravano, 2000), an iterative process of extracting relations through the use of the seed data that has high coverage (generative), but is also selective; a *distant supervision* technique (Mintz, Bills, Snow, & Jurafsky, 2009) that required a large structured database of semantic relations to govern the extraction process; and the *expectation-maximization* approach (Nigam, McCallum, Thrun, & Mitchell, 2000) that combined labeled and unlabeled data in the training of a classifier. In this approach, the classifier is first trained on only the labeled data, then it is used to estimate labels on the unlabeled data. Finally, the classifier is re-trained on both (originally labeled and labeled by the classifier) data to make predictions.

Drawing upon the concepts of snowballing and the use of unlabeled data, we designed a novel method of producing a highly generative set of training data from the corpus. We call this method the *Quality-controlled Auto-labeling*.

Quality-controlled Auto-labeling (QCAL) is a novel method of generating training data from the corpus using a *model-first* approach. In this method, a small set of seed data is used to train an empirically-chosen classifier. The classifier is then used to classify a batch of unlabeled records in the corpus. A portion of the results of the classification is reviewed and verified by a human expert at every batch. Upon approval by the human expert, the classifier is re-trained on the newly classified records as well as the original seed records. This process repeats until sufficient number of records have been labeled, as illustrated in Figure 8:

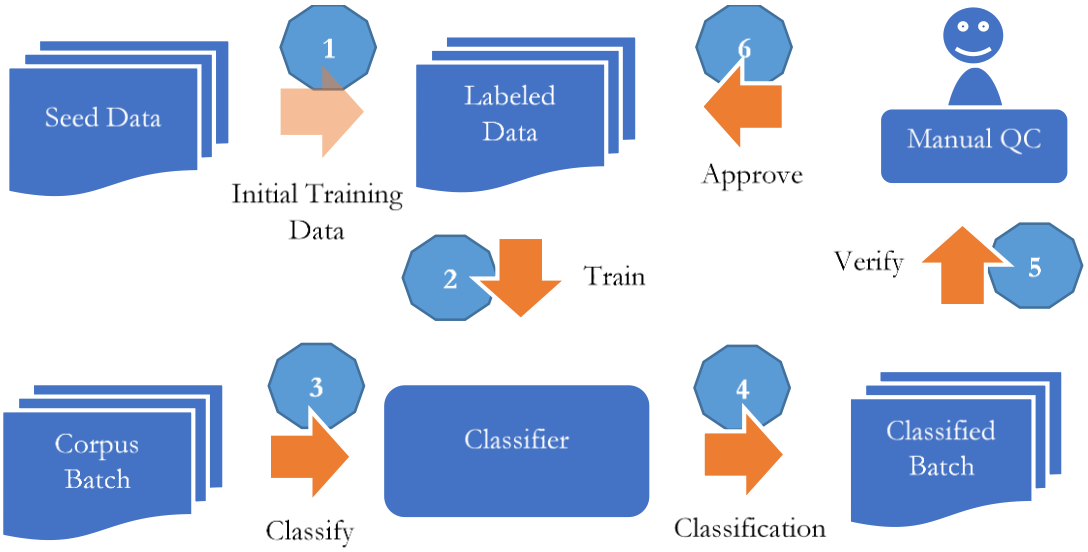


Figure 8: Quality-controlled Auto-labeling Process

3.2.4.2.1 Seed Data Generation

The seed data in the QCAL approach is highly generative and discriminative collection of records. Generating data that met these qualities required human intelligence. However, manually examining each record was not feasible in a corpus with more than 11 million records spread across 10 files. Additionally, because of the relatively low frequency of the positive records we had observed, it would be impossible to manually generate both positive and negative seed records that were representative of the corpus. We needed a better way to locate the positive seed records.

3.2.4.2.1.1 Labeling Candidate Extractor

We created a program called *Labeling Candidate Extractor* (LCE) that sequentially processed every record in the corpus looking for certain string patterns indicating a potential positive match. The string patterns were generated by the program using three lists: 186

main terms, 11 prefix terms and 21 postfix terms (See Appendix D: Positive Patterns for Seed Data for the complete list) per the following algorithm:

```
patterns = []
for main_term in main_terms:
    for prefix_term in prefix_terms:
        patterns.append(prefix_term + ' ' + main_term)

    for postfix_term in postfix_terms:
        patterns.append(main_term  + ' ' + postfix_term)
```

Figure 9: Pattern Generation Algorithm

In initial experiments, we found that LCE program took over 5 days to serially process the corpus of 11,689,482 records. To improve the processing efficiency, we redesigned the program to split each file into $n + 1$ number of chunks and parallelly process each chunk, where n is the quotient of the integer division of the total number of records in the file by 100,000:

$$n = (int)\frac{total\ number\ of\ records\ in\ file}{100000} \tag{68}$$

Each of the n chunks contained exactly 100,000 sequential records from the corpus. The last $(n + 1)^{th}$ chunk contained the remaining records (< 100,000) after records for all n chunks were allocated. We also precompiled the regular expression patterns, which reduced the time taken to process each record. With the parallelization and regular expression optimization, the LCE program completed processing the entire corpus in less than 24 hours.

The LCE program placed each record into one of four categories: Records in the corpus that matched the at least one of the patterns were considered *potential positive records*.

Records that did not match the any of the patterns and also did not have any of the main terms were considered *potential negative records*. Records that matched at least one pattern but also matched a negating prefix regular expression (NO\s+(\w+\s+)?) or negating postfix regular expression ([S]?\s+\w+\s+NOT) were considered *questionable positive records* and records that did not match any of the patterns but contained at least one of the main terms were considered *questionable negative records*. Table 11 shows the distribution of records across these categories:

Year	Potential Positive	Potential Negative	Questionable Positive	Questionable Negative
2007	8,588	8,588	83	9,999
2008	6,978	6,978	59	9,999
2009	9,777	9,777	72	10,000
2010	26,635	26,635	147	10,003
2011	6,250	6,250	330	10,000
2012	13,639	13,639	621	9,997
2013	13,654	13,654	1,392	10,000
2014	13,792	13,792	1,527	10,000
2015	25,006	25,006	1,026	10,005
2016	37,158	37,158	1,077	10,010
Total	**161,477**	**161,477**	**6,334**	**100,013**

Table 11: Results of Pattern Matching for Seed Records

Some notes may be appropriate on the contents presented in Table 11:

1. We designed the LCE program to maintain parity on the number of potential positive and potential negative records and capped both at the lower of the two. Without this strategy, the number of potential negative records would be much larger than the number of potential positive records.
2. The LCE program was also designed to extract potential negative samples close to where potential positive samples were found, instead of sequentially from the start. This was done to ensure a proper distribution of the potential negative samples across the entire file.
3. In experimental runs, we observed that the number of questionable negative records grew rapidly, so we capped those at 10,000 per file in the corpus (in some cases it's slightly higher due to their uneven distribution across $n + 1$ chunks).

3.2.4.2.1.2 Verified Sample Generator

We designed and implemented another program called *Verified Sample Generator* (VSG) to allow a human expert to disposition a set of records among those output by the LCE program as either *positive*, *negative* or *unknown*. The VSG program iteratively prompted the User to classify a randomly selected record from the pool of candidate records produced by the LCE program. The VSG program also integrated a decision support system we developed to aid the User in the classification process. Figure 10 illustrates this process of seed data generation implemented in the VSG program:

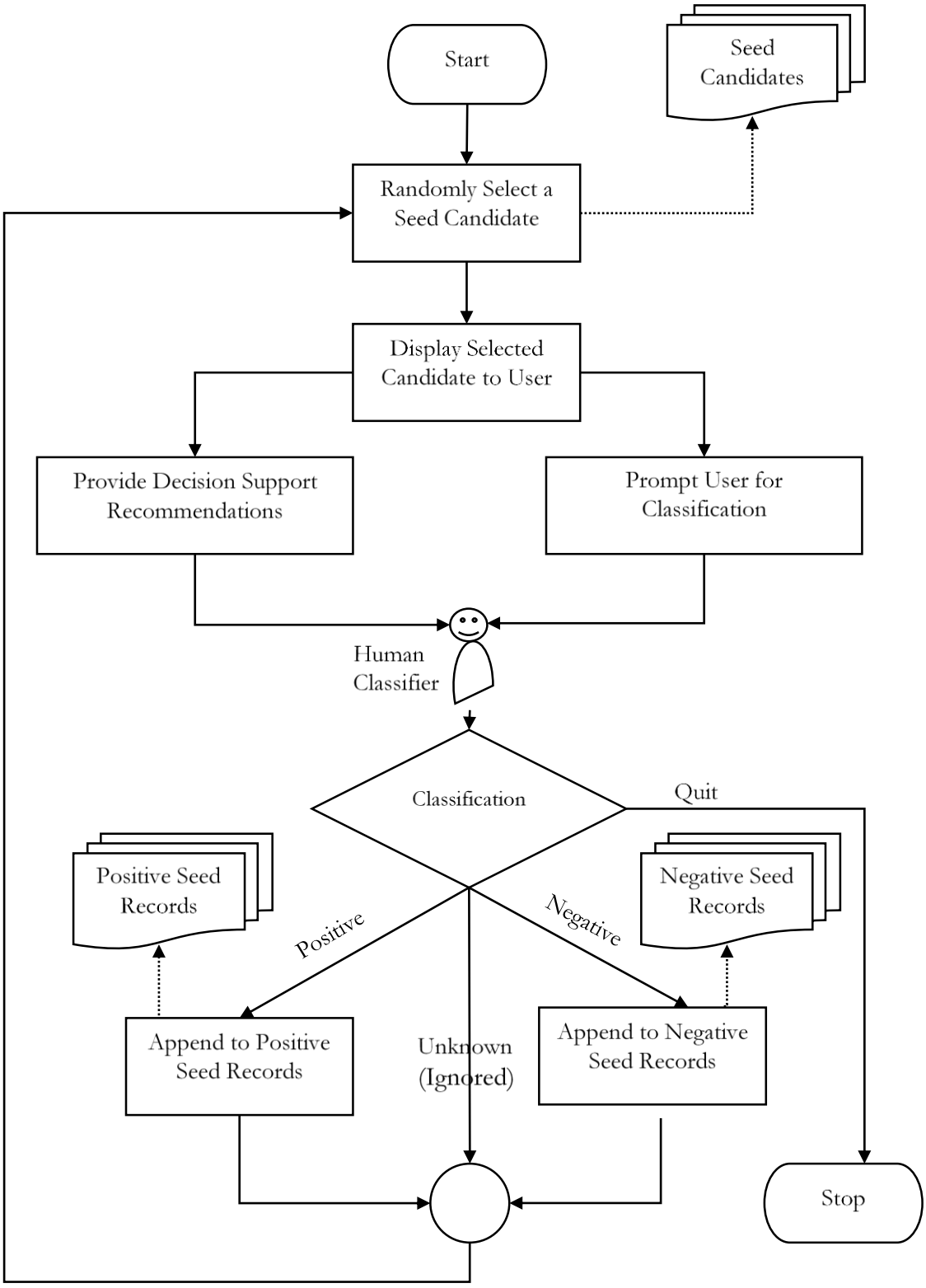


Figure 10: Seed Records Generation Process

The interactive VSG program was run over multiple sessions to build the collection of human expert-verified seeds records. Figure 11 shows the snippet of the log from a VSG program session:

Figure 11: Verified Sample Generator Program Log Snippet

After every few VSG sessions, the output (classified) records were manually reviewed and adjusted again to minimize misclassified records. The final resulting collection of seed records had 2,449 labeled narratives. Table 12 shows the yearly breakdown of these records:

Year	Seed Records
2007	151

Year	Seed Records
2008	102
2009	127
2010	204
2011	135
2012	235
2013	290
2014	335
2015	397
2016	473
Total	**2,449**

Table 12: Number of Seed Records

3.2.4.2.1.2.1 Decision Support System

To aid the user (human classifier) make correct classifications, we designed a decision support system in the VSG program that provided labeling recommendations to the classifier. The decision support system was implemented as a machine learning process itself. A total of 10 different variations of arbitrarily-selected models were trained on the available classified seed records to make recommendations on each randomly selected candidate record. As shown in Figure 11, the user was presented with suggestions from each of these models. After a set number of classifications (or the User's choice), the VSG program re-trained each of the models on the updated set of classified records.

The integration of the machine learning-based decision support system in the VSG program provided us with two key benefits:

1. It helped the human classifier make better decisions. In many occasions (usually when decision support recommendations were different than the human classifier's initial impression), the system led us to scrutinize a candidate record at a greater detail than we otherwise would have;
2. It provided us with an opportunity to empirically test the performance of various models in our use case and narrow down the list of models/classifiers we wanted to use in the bigger classification task. It was also through this process that we ruled out the bigrams and trigrams-based feature selection for our classification problem because the performance of the models using those methods of feature selection was erratic.

3.2.4.2.2 Auto Labeler

The *Auto Labeler* is a snowballing program we designed to automate the generation of labeled records through relation extraction from the corpus. This program initially used the seed records produced by the VSG program as input and generated new labeled records in batches using a machine learning technique. A portion of the newly labeled records in each batch would go through human quality control before the batch would be merged with the labeled records, which would then be used retrain the machine learning models. Figure 12 illustrates at a high level the auto labeling process we designed and implemented in the Auto Labeler program:

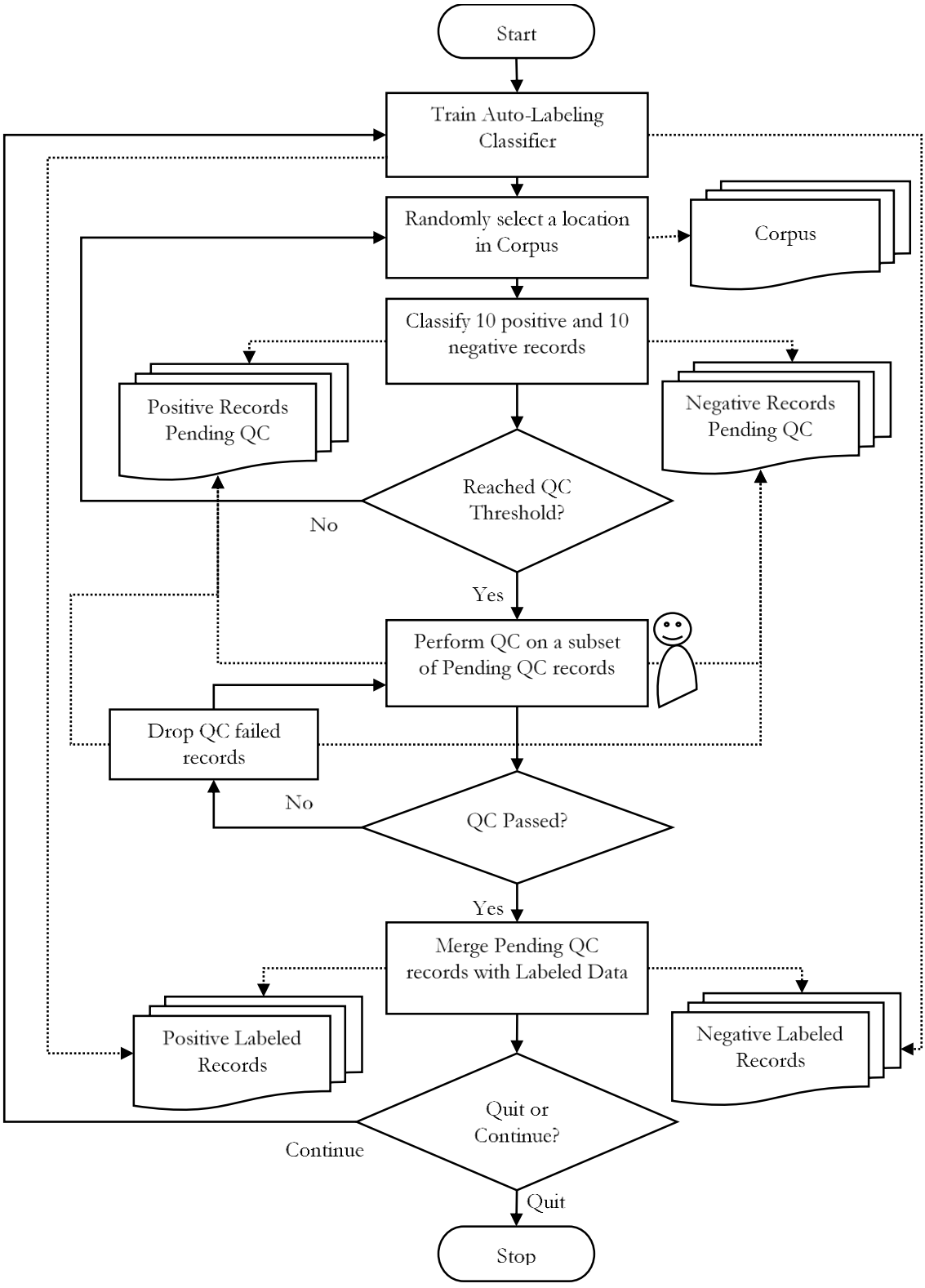


Figure 12: Auto Labeling Process

3.2.4.2.2.1 Labeling

We used a logistic regression classifier with stochastic gradient descent learning for auto labeling. The selection of this model was arbitrary but somewhat influenced by our empirical evaluation of the performance of several models during the seed data generation process described in Section 3.2.4.2.1.

However, we recognized that the output of a snowball-type system in relation extraction would be highly generative, and potentially less discriminative. We also observed that, incorporating excessively generative samples back into the training data would cause subsequent classifications to yield records with significant semantic drift.

To control for such drift, we set the auto-labeling classification threshold to 90%. In other words, the Auto Labeler would label a record as positive only if the probability of the model's prediction in the positive classification was 0.9 or more for the record. Similarly, a record would be labeled negative only if the classifier's reported probability of the prediction was at least 0.9.

To maximize the sampling distribution of selected records across the corpus, the Auto Labeler program sourced records from multiple files in the corpus, and also multiple sections in the same file. It also maintained a list of previously selected records so that they would not be selected again. This labeling process (decomposition of process "Classify 10 positive and 10 negative records" in diagram Figure 12) is illustrated in Figure 13:

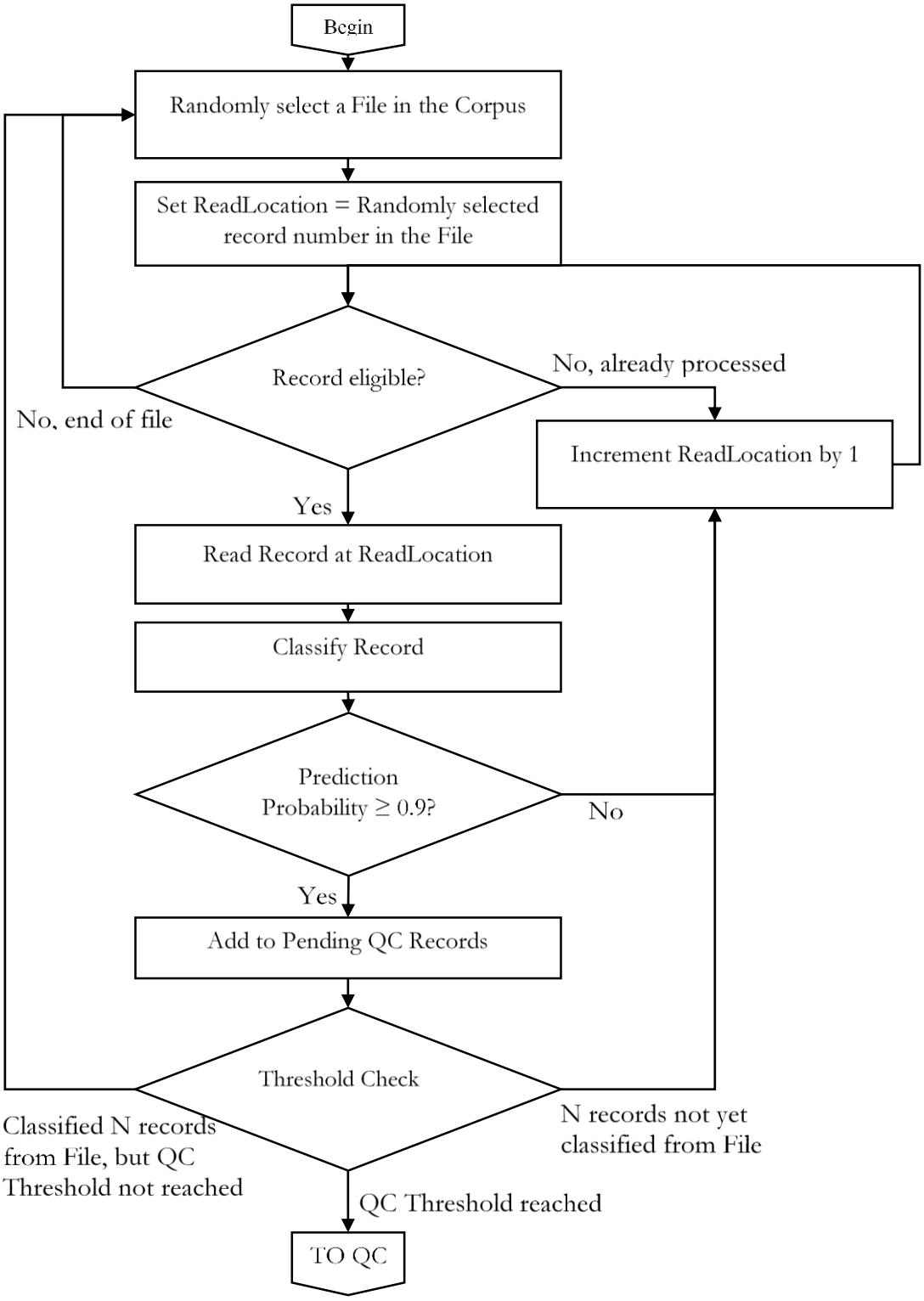


Figure 13: Labeling Process Loop

3.2.4.2.2.2 Quality Control

A central and novel element of our design of Auto Labeler was the Quality Control (QC) system that was built in. The Auto Labeler program, as described in previous sections, performed labeling in batches. In our experiments, each batch was made of m randomly selected records from multiple files in the corpus, where m was initially set to 10, but gradually increased to an arbitrarily much higher value as the labeled data built up. At the end of every batch, labeled (but pending-QC) records would go through a QC process where a random subset of the records would be verified by a human expert using the interactive QC feature of the Auto Labeler program. If all of the records in the subset passed QC (i.e. human confirmed the subset as accurately classified), then the batch would be merged with the labeled records. But if the human expert corrected any of the classifications during the QC process, then the QC process would repeat with a new subset of pending-QC records until the human expert confirmed the QC as passed. The number of samples in the subset to verified was initially very high (near 100% of the records pending QC), but we gradually lowered it (to 5%) as we established confidence in the process.

In addition to the batch-level QC using the Auto Labeler program, we also performed manual reviews and necessary adjustments of the resulting labeled records. To do this, at the end of every few Auto Labeler sessions, we examined the sorted set of all auto labeled records by the probability value associated with each record's classification (as positive or negative) and manually reviewed and eliminated records that had relatively low probability and did not, in the human expert's judgment, belong in the correct class. We also

manually reviewed records at random locations throughout the training data frequently to remove any obviously misclassified records from both positive and negative training sets.

3.2.4.2.2.3 Controlling Semantic Duplicates

During the course of our research, we observed that Medical Device Reporting submitters often used identical verbiage to describe multiple instances of the same issue. Consider the following three records, for example:

932750|691038|D|1||A CUSTOMER REPORTED AN ISSUE WITH THEIR FREESTYLE METER. UPON PROD INVESTIGATION, IT WAS DISCOVERED THE METER EXHIBITED THE MEMORY OVERWRITE MALFUNCTION.

911174|683853|D|1||A CUSTOMER REPORTED AN ISSUE WITH THEIR FREESTYLE METER. UPON PROD INVESTIGATION IT WAS DISCOVERED THE METER EXHIBITED THE MEMORY OVERWRITE MALFUNCTION.

925888|721562|D|1||A CUSTOMER REPORTED AN ISSUE WITH THEIR FREESTYLE METER. UPON PROD INVESTIGATION IT WAS DISCOVERED THE METER EXHIBITED THE MEMORY OVERWRITE MALFUNCTION.

While the Auto Labeler program rejected entirely duplicate records, narratives like the ones listed above were initially treated as unique because they belonged to different reports. However, it became clear to us that without any controls, our labeled data sets would be overwhelmingly dominated by these *semantically* duplicate (although technically unique) records because of their frequency in the corpus. Such flooding would leave other records of interest out of our labeled dataset and could have potential downstream impact of skewing our machine learning models. Therefore, we implemented a semantic duplicate record check feature to the Auto Labeler program, which allowed only up to a configured number of identical narratives to be included in the labeled datasets. For our research, we set the value of this configuration parameter to 10.

3.2.4.2.2.4 Number of Auto-Labeled Records

After several Auto Labeler sessions and manual adjustments, we generated a total of 72,726 positive and 72,975 negative auto-labeled records. These records were distributed across all 10 files in the corpus. Table 13 shows the distribution of the auto-labeled sample records across the corpus for each year.

Year	Negative Samples	Positive Samples	Total Samples	Percent of Corpus
2007	3,951	4,198	8,149	3.5%
2008	3,790	4,073	7,863	2.97%
2009	3,807	3,785	7,592	1.96%
2010	3,852	4,112	7,964	1.14%
2011	4,812	5,892	10,704	1.1%
2012	5,928	7,537	13,465	1.08%
2013	7,661	8,595	16,256	1.06%
2014	12,022	11,338	23,360	1.19%
2015	13,648	11,708	25,356	1.11%
2016	13,504	11,488	24,992	1.19%
Grand Total	**72,975**	**72,726**	**145,701**	**1.25%**

Table 13: Number of Auto-Labeled Records

3.2.5 Trained Model Generation

It should be noted that not all of the auto-labeled records were verified by a human expert for the accuracy of the assigned labels. While we applied extensive care in maximizing

their accuracy through QC and manual reviews as discussed in the previous section, it was possible that some of the records in this set may still have been misclassified.

To train our computer models, our options were to either: a) Use an entirely human-verified but very small set of seed records; or b) Use a much broader set of training data generated through a snowballing technique, that was only partially human-verified at the record level, but quality-controlled through multiple processes discussed in the previous section. After a series of experimental runs, we decided to pursue the latter option (b) because it provided us with the training data that was more representative of the corpus and helped us accurately classify significantly more records. To control for potential semantic drifts or the effects of potential misclassified records in the training data, we opted to make our classification (see Section 3.2.6) and verification (Section 3.4) designs more rigorous and adjust the training data and parameters until the overall classification performance was acceptable.

As explained in Section 3.2.3, we identified three classifiers as suitable candidates for our classification experiments: Naïve Bayesian, logistic regression and support vector machine with stochastic gradient descent learning. We designed and implemented a program called *Trained Model Generator* (TMG) to train each of these classifiers using the labeled records generated by the Auto Labeler program discussed in Section 3.2.4.2.2. The TMG trained each Classifier using a sequence of processes as illustrated at a high level in Figure 14:

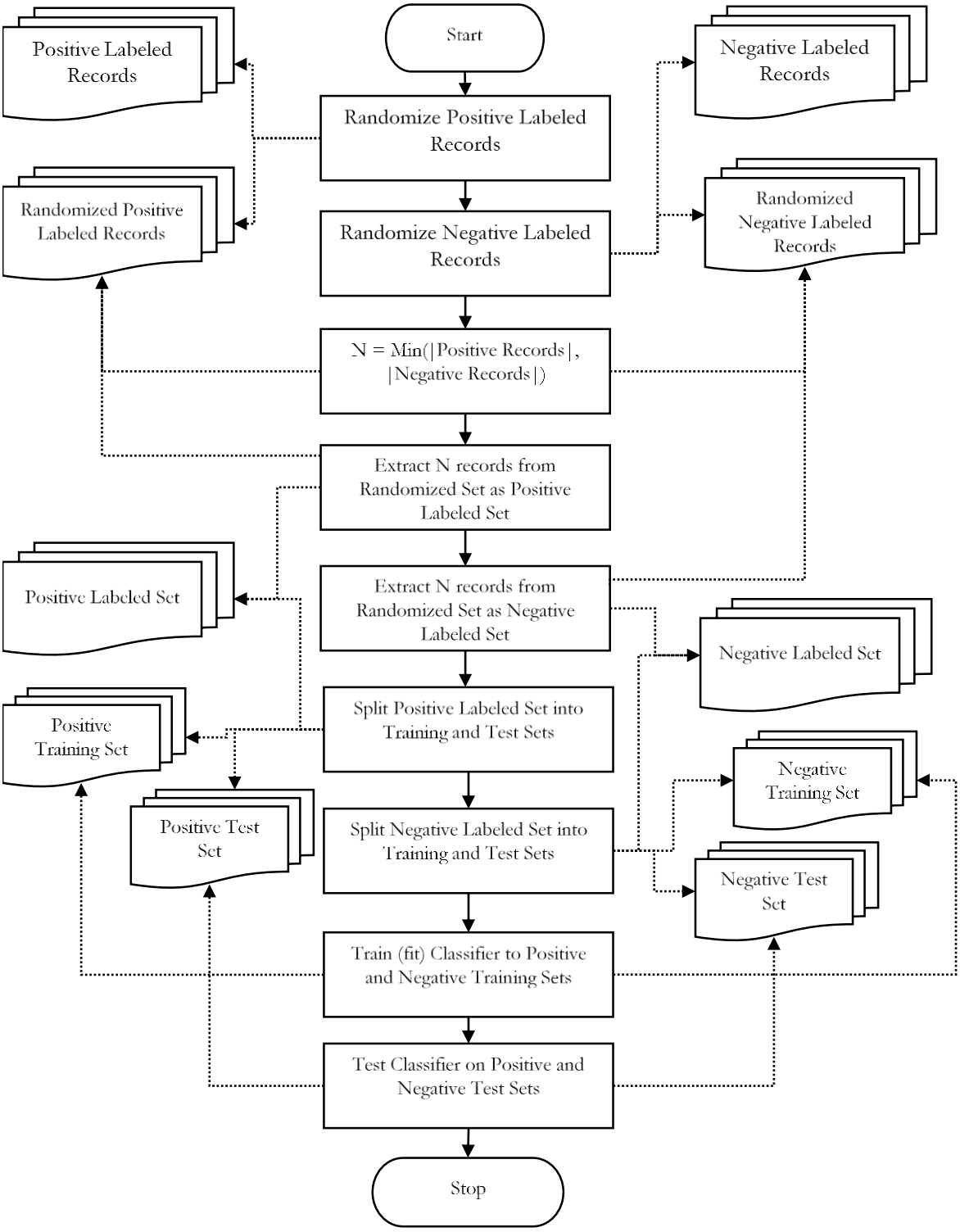


Figure 14: Trained Model Generation Process

At a high level, the randomized set of the labeled records were split into training and testing sets. The training set was used to train the model, while the testing set was used to test the accuracy of the trained model. In the sections below, we will discuss some key elements of the TMG design:

3.2.5.1 Randomization

In order to ensure random distribution of records in the training data, the TMG program shuffled all (72,726 positive-labeled and 72,975 negative-labeled) records produced by the Auto Labeler program and stored the randomized sets for further processing.

3.2.5.2 Training and Test Sets

The TMG program ensured equal number of positive and negative records in the training data. To achieve this, it extracted N records from each of positive and negative labeled sets, where:

$$N = \min(|positive\ records|, |negative\ records|) \tag{69}$$

The N = 72,726 records from each set were then divided into two sets: *training set* to be used in training the model; *test set* to be used in testing the model's accuracy. Table 14 describes our allocation of the records into train and test sets:

	Positive Records	**Negative Records**	**Total**
Training Set	54,544	54,544	**109,088**
Test Set	18,182	18,182	**36,364**
Total	**72,726**	**72,726**	**145,452**

Table 14: Train and Test Sets

3.2.5.3 Feature Extraction

In order to train (*fit*) the classifiers, we extracted a vocabulary of words found across all 109,088 records in the training set and built a sparse matrix of counts for each word (*token*). The TMG program utilized the *CountVectorizer* class in Python's *scikit-learn* module (Pedregosa et al., 2011) to build this matrix. We did not pre-create a dictionary of features of our interest; instead, we allowed almost every word (token) in the records to be a feature (some exclusions are discussed in sections below). Then we transformed the matrix of count vectors to a normalized matrix that adjusted for the size of each record. We describe key elements of our feature extraction process in the sections below:

3.2.5.3.1 Minimum Document Frequency

For a token in the vocabulary to be a feature, we required that the token be present in at least 0.10% (one-tenth of a percent) of records. This cutoff was set to contain the number of highly infrequent terms in the list of features.

3.2.5.3.2 Stop Words

After several experimental classification runs leading up to the selection of our models, we discovered that the training data contained a number of highly frequent but non-discriminative terms. To minimize the bias of these terms in our experiments, we excluded a total of 569 terms from the list of features. Among these, 318 were commonly occurring terms in the English language and the remaining 251 were terms we identified as noise through our series of experiments. See Appendix E for a complete list of these stop words.

3.2.5.3.3 TF-IDF Transformation

The simple count-based approach of feature extraction (such as through a count matrix discussed earlier) suffers from a major limitation: It can potentially give larger-size records higher weight than they deserve. This is because the frequency of a feature term in a record could be proportional to the size of the record: Records that are larger may have a higher frequency of a given feature term, while smaller records may have lower frequency of the term even though the actual discriminative significance is the same in both cases. So, to control the bias of larger size records in the training of the model, we performed a transformation of the count matrix to a normalized matrix that takes into account the size of the records, as well as the significance of a feature term using the *term-frequency – inverse-document-frequency* (TF-IDF) algorithm.

The TMG utilized the *TfidfTransformer* class of the Python scikit-learn module (Pedregosa et al., 2011) to compute the TF-IDF matrix. Formally, the scikit-learn implementation of the TF-IDF algorithm for a given record (also known as document) d and feature term t in a set of records $d \in D$ can be described as:

$$tfidf(t, d, D) = tf(t, d) \times idf(t, D) \tag{70}$$

Where $tf(t, d)$ is the frequency of the term t in record d; and $idf(t, D)$ in our use case can be defined as:

$$idf(t, D) = \log_e \left(\frac{1 + |D|}{1 + df(t, D)} \right) + 1 \tag{71}$$

Where, $df(t, D)$ is the number of records in the training set D containing term t. Note, in simple terms, $idf(t, D)$ is essentially a ratio of total number of records in the training set to the number of records containing the term t, on a logarithmic scale. The

adjustments related to the addition of 1 in the nominator and the denominator are to prevent error conditions such as division by zero.

3.2.5.3.4 Features

With the adjustments discussed in the previous sections applied, the number of features extracted from the 109,088 records in the training set were slightly variable across models (due to randomization discussed in Section 3.2.5.1), but less than 1,900. Table 15 lists the number of features by model:

Naïve Bayesian	Logistic Regression	Support Vector Machine
1865	1868	1860

Table 15: Number of Features by Model

3.2.5.4 Classifier Training

We then created a set of classifiers each representing the selected models and trained them using the corresponding TF-IDF matrix and associated labels. The TMG program utilized the following classes from Python's *scikit-learn* module (Pedregosa et al., 2011) to instantiate the classifier objects (See Appendix F for the complete list of various parameters used):

Model Name	Scikit-learn Classifier Class	TF-IDF Matrix Shape
Naïve Bayesian	sklearn.naive_bayes.MultinomialNB	109088 × 1865
Logistic Regression	sklearn.linear_model.LogisticRegression	109088 × 1868

Support Vector Machine with Stochastic Gradient Descent Learning	sklearn.linear_model.SGDClassifier	109088 × 1860

Table 16: Scikit-learn Classifiers

3.2.5.4.1 Most Informative Features

To verify that each classifier's training was reasonable, we obtained a list of most informative features for each of the positive and negative classes. Table 17 lists the 25 most informative positive features reported by each classifier (See Appendix G for a longer list of most informative positive *and negative* features):

Multinomial Naïve Bayes	**Logistic Regression**	**Support Vector Machine with Stochastic Gradient Descent**
ISSUE	SOFTWARE	ERROR
ERROR	ERROR	SOFTWARE
EVENT	DISPLAY	DISPLAY
DISPLAY	ALARM	ALARM
ALARM	SCREEN	SCREEN
DEVICE	BOARD	ISSUE
INDICATION	HISTORY	BOARD
METER	BOOT	HISTORY
KEYPAD	KEYPAD	BOOT
BUTTON	BUTTON	BUTTON

Multinomial Naïve Bayes	**Logistic Regression**	**Support Vector Machine with Stochastic Gradient Descent**
TESTED	ISSUE	DISPLAYED
SOFTWARE	COMPUTER	METER
REPORT	IMAGE	IMAGE
USER	ALARMS	MONITOR
SCREEN	DISPLAYED	KEYPAD
ANALYSIS	MEMORY	COMPUTER
BATTERY	BUTTONS	ALARMS
EVALUATION	ALARMED	TESTED
INFORMATION	TOUCHSCREEN	TOUCHSCREEN
MESSAGE	MESSAGE	ALARMED
STATED	MONITOR	SHUT
DISPLAYED	PROGRAMMING	MESSAGE
TIME	LOCKED	ENGINEER
MEDICAL	METER	PROGRAMMING
DELIVERY	LOG	STATED

Table 17: Most Informative Features

3.2.5.5 Classifier Scores

After training, each of the classifiers was tested against 36,364 records in the test set. We obtained the score of the classifier as a measure of its accuracy based on its classification of the records in the test set. Table 18 shows the accuracy of each of the classifiers:

Classifier	Accuracy
Multinomial Naïve Bayes	0.97
Logistic Regression	0.98
Support Vector Machine with Stochastic Gradient Descent	0.98

Table 18: Classifier Accuracy

3.2.5.6 Classifier Persistence

The TMG program saved to disk each of the trained classifiers in a binary serialized form. This allowed the classifiers to be portable and used in different sessions or programs. Our classification program discussed in the next section utilized these persisted classifiers in the classification tasks without having to retrain them.

3.2.6 Classification

Using the models and trained classifiers generated by the TMG program, we classified each of the 11,689,482 narrative records across 10 files in the corpus. To perform this classification, we designed and implemented a program called *Classifier*, which implemented a unique classification scheme. We discuss the design of this program and the parameters of our use in the sections below.

3.2.6.1 Classification Scheme

The Classifier program sequentially processed each file in the corpus. For each input file, it classified all records in the file and produced a pair of files (per classifier) containing positive and negative records respectively. The program also generated a pair of positive and negative records file containing the results of the overall classification. The results of the classification (classifier-level, as well as overall) were summarized in yet another file. Figure

15 illustrates the high-level corpus classification scheme we implemented in the *Classifier* program:

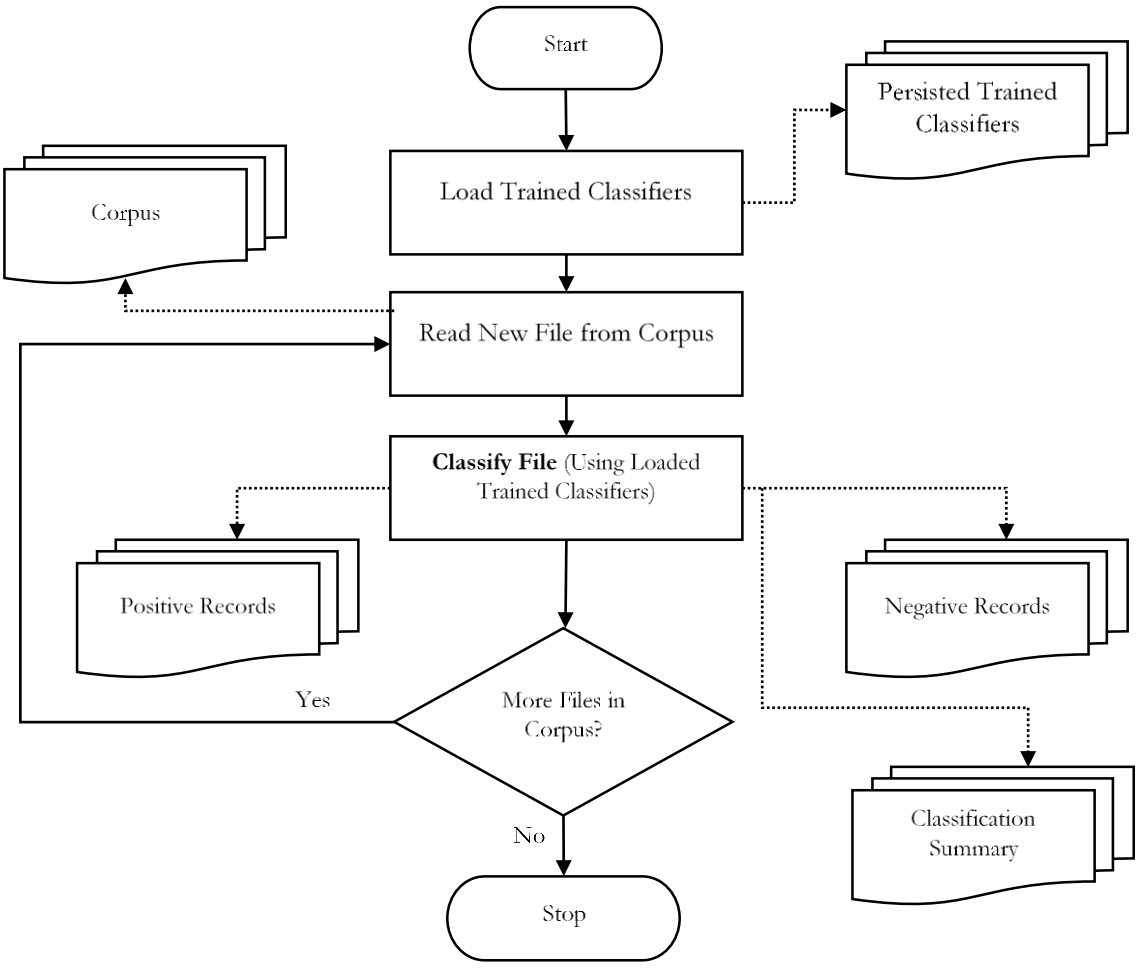


Figure 15: Corpus Classification Scheme

For each file in the corpus, the *Classifier* program sequentially processed every record and classified it using each of the three trained classifiers. Figure 16 illustrates the high-level design of the corpus file classification scheme we implemented (expanded view of Classify File process in the flowchart in Figure 15):

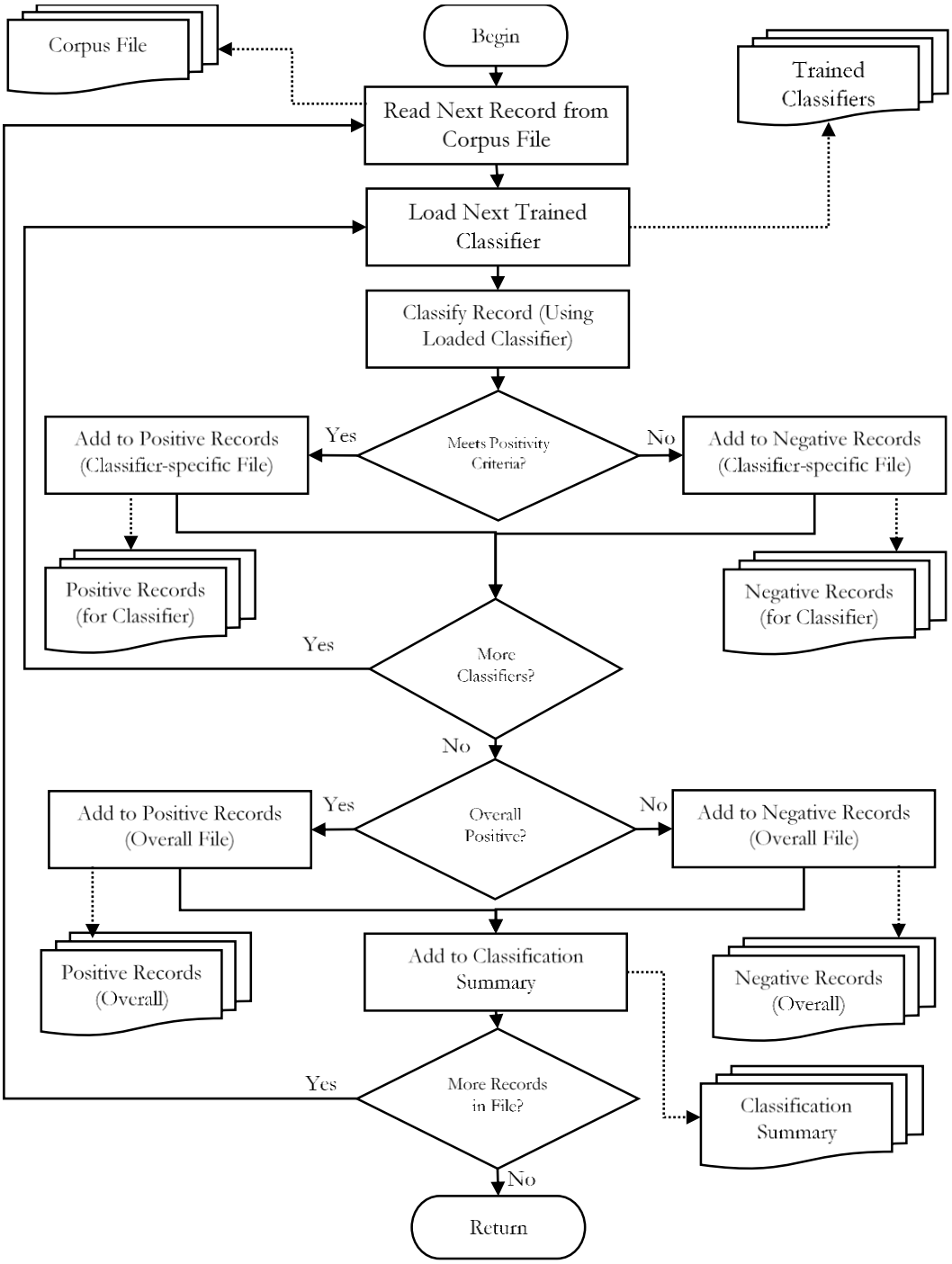


Figure 16: Corpus File Classification Scheme

3.2.6.1.1 Positivity Criteria

On the result of the binary classification made by each classifier based on its learning, we applied additional criterion to qualify a record as positive. Specifically, for each model, we required the following three conditions to be satisfied for a record to be classified positive:

1. The record is at least 40 characters in length;
2. The classifier has classified the record as positive; and
3. The classifier's reported probability in the classification is at least 0.95

We added the minimum length check based on our empirical observations that any record less than 40 characters long (note, first 22 characters are non-narrative data such as keys) could not contain reliable signals for classification and could be ignored. The probability check was added mainly for two reasons:

1. Establish a higher level of confidence in the classification; and
2. Control for any semantic drift in a snowball-type relation extraction method we implemented in the training data generation process.

It is important to note the Support Vector Machine-based classifier is not a probabilistic predictor. Therefore, the minimum probability requirement was not applicable for classifications made by this classifier.

3.2.6.1.2 Ensemble Positivity Criteria

The *Classifier* program enforced the Positivity Criteria discussed in the previous section for each classifier on each record. The result of this classification was an assignment of a label (positive or negative) to each record in the context of each classifier.

We then took the results of the classification by all three models for each record and evaluated whether the record could be considered *overall* positive. To make this overall determination, we followed a unanimous vote approach: For a record to be considered *overall* positive, *all three* classifiers in the ensemble were required to classify it as positive using the positivity criteria discussed in the previous section.

3.2.6.2 Detecting Potential False Negatives

To assess the effectiveness of our classification scheme in a series of experiments leading up to the final design of the *Classifier* program, we sampled a small set of classified records at various segments and performed a manual review. This method (manual sampling) worked reasonably well to detect potential false positive records but did not work well to detect false negative records.

The method worked well to detect false positive records because the positive classified records were a much smaller set of the corpus and the sample generally represented the make-up of the positive records set. For example, if we sampled 100 records from various positions in the positive records set, we found that the resulting sample would generally describe the positive records set. However, if we followed the same approach to detect potential false negative records, we found that our sampling missed positive records in the negative records set. This was because our likelihood of drawing a positive record from the much vaster set of negative classified records was too low. For this reason, we felt that the manual sampling approach to assess the false negative rate of our classification would not be adequate.

As a way for detecting potential false negative records, we devised and implemented a strategy in the final design of the *Classifier* program: During classification of each record,

the program would also scan the record checking for the presence of any of the 186 positive pattern terms that we identified in the Seed Data Generation process (See Section 3.2.4.2.1). If a record was classified as *negative* but contained at least one positive pattern term, the Classifier flagged that record as a potential false negative. This approach had the potential of exaggerating the false positive rate (not all records containing the pattern term may indeed be positive), but it provided us with a way of measuring the effectiveness of our machine-learning based-approach. The results of this technique are discussed in Section 3.4.3.

3.2.6.3 Classification Summary Report

Upon classification of each record, the *Classifier* program generated a classification summary, which contained high-level results of the classification task for the record. For each record classified, the classification summary contained:

- Record Identifying Information
- Classifier Identifier
    - 'sklearn.mnb' for Multinomial Naïve Bayes
    - 'sklearn.sgd' for Support Vector Machine with Stochastic Gradient
    - 'sklearn.logreg' for Logistic Regression
    - 'overall' for overall classification
- Classifier's Reported Probability of Positive Classification
- Negative Probability
    - For this, we simply subtracted the positive probability from 1.
- Classification
    - 'pos' or 'neg'
- Positive Pattern Term (if any found)

3.3 MAPPING WITH MAUDE EVENTS

Once all 11,689,482 narrative records were classified using the machine learning technique described in Section 3.2, we imported the summary of the classification into the SQL Server database for analysis. Using the record identifiers in the summary report, we mapped the classification result records to the event information records in the MAUDE events table. This allowed us to perform advanced queries across entities generated through the classification process as well as the original MAUDE datasets.

After we had imported the classification summary, we verified that the import process had captured the overall classification of all narrative records. Table 19 shows the yearly breakdown of the classification summary:

Year	Classification		Total Number of Records
	Negative	Positive	
2007	204,941	27,685	232,626
2008	234,772	30,199	264,971
2009	350,365	37,676	388,041
2010	618,245	79,227	697,472
2011	885,984	86,496	972,480
2012	1,141,467	110,053	1,251,520
2013	1,396,041	140,421	1,536,462
2014	1,692,794	272,264	1,965,058
2015	1,857,840	416,247	2,274,087

Year	Classification		Total Number of Records
	Negative	Positive	
2016	1,775,659	331,106	2,106,765
Grand Total	**10,158,108**	**1,531,374**	**11,689,482**

Table 19: Overall Classification Summary

The overall summary showed that of the 11,689,482 narrative records, 1,531,374 (13%) were positive (i.e. related to computing technology). However, this did not mean that 13% of all MAUDE events reported in the same period were related to computing technology. This is because each event report could contain multiple narrative records. On average, the 11,689,482 narrative records across 5,110,200 events (See Section 1.2.1.2.1) translated to 2.29 narrative records per event.

3.4 VERIFICATION

Our machine learning experiments were iterative: We ran and adjusted the models and our algorithms repeatedly until we felt comfortable with their performance. After the models and algorithms were finalized and the classification performed, we assessed the outcome of the classification in multiple ways: First, we calculated a set of metrices standard in any machine learning task: precision, recall and f-1 scores. Second, we manually evaluated a sampling of positive classified records for the presence of negative records. We also implemented an automated approach to identify potential positive records in the set of negative classified records. The sections below describe our process and the results in more detail.

3.4.1 Classification Metrics

A routine method of assessing the performance of a machine learning-based classification task is through a set of statistical measures. These measures include the Confusion Matrix, Precision, Recall and F-1 Scores. Each of these measures assesses the performance of the model's predictions on a set of pre-labeled records. For our analysis, we used the human-verified seed data (see Section 3.2.4.2.1) to benchmark the performance of the ensemble-based, overall classification. We discuss each of these measures in the section below.

3.4.1.1 Confusion Matrix

A confusion matrix is a table that summarizes the performance of a classifier in terms of the accuracy of its predictions. Table 20 shows this summary for our ensemble-based overall classification scheme:

n=2,449	**Predicted Negative**	**Predicted Positive**	**Total**
Actual Negative:	1,186 *(True Negative or TN)*	34 *(False Positive or FP)*	**1,220**
Actual Positive:	165 *(False Negative or FN)*	1,064 *(True Positive or TP)*	**1,229**
Total	**1,351**	**1,098**	**2,449**

Table 20: Confusion Matrix

3.4.1.2 Precision

The precision of a classification is defined as the ratio of true positive records to all positive classified records:

$$P = \frac{TP}{TP + FP} \tag{72}$$

Using Table 20, the precision of our classification is:

$$P = \frac{1{,}064}{1{,}098} = 0.97$$

This indicates that, of all records we classified as positive, 97% were indeed positive. This is consistent with the goal of our classification where we prioritized minimizing the false positive rate.

3.4.1.3 Recall

The recall of a classification is the ratio of true positive records to all actual positive records:

$$R = \frac{TP}{TP + FN} \tag{73}$$

Using Table 20, the precision of our classification is:

$$R = \frac{1{,}064}{1{,}229} = 0.87$$

This implies that, of all records that were actually positive, our we classified 87% as positive. This is also consistent with our design goal of allowing a higher rate of false negative records in favor of minimizing the rate of false positive records. It is for this reason that we designed our classification as a unanimous consensus-based model ensemble, in which a record was classified as positive only if all constituent models classified it as positive, and with a probability confidence in the classification of at least 0.95, where supported.

3.4.1.4 F_1 Score

The F_1 score is a measure of the overall accuracy of a binary classification. This measure incorporates both precision (P) and recall (R) through a harmonic mean:

$$F_1 = 2\,\frac{P \times R}{P + R} \tag{74}$$

Using, (72) and (73) the F_1 score of our classification is:

$$F_1 = 2\,\frac{0.97 \times 0.87}{0.97 + 0.87} = 0.92$$

It should be noted that for simplicity, this computation uses pre-rounded values for the parameters. Therefore, the result may be slightly less precise than if the parameter values were raw. More precise values computed from the raw parameters are presented in Table 21.

3.4.1.5 Constituent Model Metrics

In the previous sections, we discussed the performance metrics for the model ensemble-based classification scheme. Since the classification of a narrative record in our experiment used the ensemble technique, the metrics discussed in Sections 3.4.1.1 through 3.4.1.4 represent the performance of our overall classification. In Table 21, we present the performance metrics for each of the three constituent classifiers against the same test data (N=2,449).

Model	**TP**	**FP**	**TN**	**FN**	**Precision**	**Recall**	**F_1 Score**
Naïve Bayes	1,114	58	1,162	115	0.95	0.91	0.93
Logistic Regression	1,089	52	1,168	140	0.95	0.89	0.92

Model	TP	FP	TN	FN	Precision	Recall	F_1 Score
Support Vector Machine with Stochastic Gradient Descent Learning	1,202	202	1,018	27	0.86	0.98	0.91
Overall (Ensemble)	**1,064**	**34**	**1,186**	**165**	**0.97**	**0.87**	**0.91**

Table 21: Classifier Performance Metrics

As shown in Table 21, each of the models we used in our classification had an F_1 score of 0.91 or higher. Note that the precision for the Support Vector Machine-based classifier is significantly lower than that of the other two classifiers in the ensemble. The reason for the lower value is because the classifier did not report a probability value on its classifications for us to set a minimum probability threshold (See Section 3.2.6.1.1 for more information). This resulted in a larger number of false positive records than the other two classifiers. However, we mitigated this issue by designing the ensemble as a unanimous voting classifier. The effectiveness of this mitigation is shown in the metrics for the overall (ensemble) classifier.

3.4.2 Manual Review of Random Positive Records

In addition to the metrics on the performance of our classification, we also took a random sampling of 1,000 positive classified records and performed a manual review. This sampling was done using a T-SQL method of generating and sorting by randomly-generated globally unique identifiers.

In our manual review of the 1,000 records, we found that 38 were actually not positive. This translates to a positive precision rate of 96.20%, roughly in line with the

precision value (0.97) we obtained through the Confusion Matrix discussed in Section 3.4.1.2.

For the manual review, we considered a positive classified record as false positive if the description in the narrative did not meet any of the following conditions:

- Event is possibly related to the computing technology onboard the problem device, including:
    - Computers and accessories (keyboards, keypads, mouse, joysticks, display monitors, disks, removable media, etc.);
    - Computer components (integrated circuit boards, storage, digital sensors, meters, transmitters, receivers, etc.)
    - Microcontrollers, microprocessors
    - Software, firmware
    - Internet, networking apparatus
    - Electronic files, records
    - Wired or wireless digital communication
- Onboard computing technology has a (primary or secondary) role in the generation, detection or presentation of the event:
    - System errors
    - Alarms, and alerts
- Onboard computing technology is an area of investigation in response to the event:
    - Log files, memory dumps, etc.

- Suspicion is sufficient; not confirmation (it is not possible to establish confirmation)

It is important to note that during the manual review, we did not seek to conclusively adjudicate the classification in cases of conflicting information. Instead, we assumed the reporter to be correct, even when the follow up investigation may have been unable to reproduce the reported scenario. For example, in the following narrative, the investigator is unable to reproduce an alarm. We, nevertheless, treat this as a *positive* record because we assume the reporter to be correct, even if the follow up did not confirm the reporter's observation:

> *…RECEIVED AND EVALUATED THE DEVICE IN QUESTION. THE EVAL WAS ABLE TO CONFIRM THE SYSTEM ERROR 322 ALARM THROUGH REVIEW OF THE DEVICE EVENT HISTORY LOG. THE ALARM COULD NOT BE REPRODUCED AND A CAUSE COULD NOT BE IDENTIFIED. THE PUMP WAS TESTED WITH PASSING RESULTS. BAXTER HAS INITIATED AN APPROVED REMEDIAL ACTION; HOWEVER THIS DEVICE HAS A VERSION OF SOFTWARE THAT DOES NOT HAVE AN APPROVED SOFTWARE UPDATE. THE DEVICE WAS FOUND TO BE OPERATING MECHANICALLY AS DESIGNED AND NO ACTION WILL BE TAKEN AT THIS TIME…*

We also did not verify the accuracy of the error messages, alarms or alerts (it is not possible verify). We treated each manifestation of the error message, alarm or alert on a computerized system/product as related to computing-technology. For example, we treated the following narrative as positive because we know that the insulin pump is a computerized product:

> *…CUSTOMER RECEIVED A MOTOR ERROR ALARM. CUSTOMER DID NOT REPORT A MOTOR POSITION ENCODER ERROR ALARM. CUSTOMER'S BLOOD GLUCOSE WAS 400 MG/DL. DURING TROUBLESHOOTING FOR MOTOR ERROR ALARM, IT WAS FOUND THAT INSULIN PUMP WAS NOT EXPOSED TO MAGNETIC FIELD OR MRI; DRIVE SUPPORT CAP APPEARED NORMAL AND CUSTOMER*

THINKS WAS ABLE TO COMPLETE REWIND PROCESS. ALARM HISTORY SHOWED ONE MOTOR ERROR ALARMS WITHIN THE LAST MONTH AND NO MOTOR POSITION ENCODER ERROR ALARM. INSULIN PUMP PASSED DISPLACEMENT TEST. CUSTOMER WAS ADVISED TO MONITOR INSULIN PUMP…

3.4.3 Automated Checks for Potential False Negatives

As discussed in Section 3.2.6.2, we tagged a record as potential false negative if the record did not meet the positivity criteria but contained at least one positive signal term. Of the 10,158,108 records classified as overall negative, 56,8201 (or 5.59%) contained at least one positive signal term. However, we also found that in many cases, these records were indeed negative despite the presence of the positive signal terms. A small casual sampling of these records is listed below (signal word highlighted):

*912361|8128187|N|1||EXAMINATION WAS NOT POSSIBLE, AS THE DEVICES WERE NOT RETURNED. A COMPLAINT **DATABASE** SEARCH FINDS NO OTHER REPORTED INCIDENTS AGAINST THE KNOWN PRODUCT AND LOT CODE SINCE RELEASE FOR DISTRIBUTION. THE INVESTIGATION COULD NOT VERIFY OR IDENTIFY ANY EVIDENCE OF PRODUCT CONTRIBUTION TO THE REPORTED EVENT. BASED ON THE INVESTIGATION, THE NEED FOR CORRECTIVE ACTION IS NOT INDICATED. SHOULD ADDITIONAL INFORMATION BE RECEIVED, THE COMPLAINT WILL BE REOPENED. DEPUY CONSIDERS THE INVESTIGATION CLOSED.*

*810952|560359|D|1||IT WAS REPORTED THAT A **FILE** SEPARATED IN THE CANAL DURING A PROCEDURE. THE DOCTOR PLANS TO REFER THE PATIENT TO A SPECIALIST, THOUGH OUTCOME OF THE EVENT IS NOT KNOWN AS OF THIS REPORT.*

*870597|19003866|N|1||DEVICE EVALUATION: THE DEVICE IS CURRENTLY BEING EVALUATED; THE MANUFACTURER WILL **FILE** A FOLLOW-UP REPORT DETAILING THE RESULTS OF THE EVALUATION ONCE IT IS COMPLETED.*

*896437|17369482|D|1||THE CUSTOMER REPORTED THAT THE ORTHO PROVUE ANALYZER DRIPPED FLUID DURING TYPE AND **SCREEN** TESTING AND THAT THE PROBE WAS BENT. PROBE DRIP MAY LEAD TO DILUTION OF SAMPLE / REAGENT, CARRY OVER AND / OR CROSS*

CONTAMINATION AND ERRONEOUS RESULTS WHICH COULD LEAD TO TRANSFUSION OR INCOMPATIBLE BLOOD. ERRONEOUS TEST RESULTS WERE NOT REPORTED.

*857358|7967243|N|1||NO CONCLUSIONS CAN BE MADE AT THIS TIME. NO **CONNECTION** CAN BE MADE BETWEEN THE REPORTED EVENT AND ANY SHORTCOMING OF THE DEVICE AT THIS TIME. DEVICE IS APPROVED AS SINGLE LEVEL IMPLANT 2-LEVEL INSERTION IN AN OFF LABEL USE OF THE DEVICE.*

While this method may not be any more precise in assessing the accuracy of our classification than the metrics discussed in Section 3.4.1, the results of this exercise reinforced the merits and effectiveness of using a machine learning-based approach for the classification task, compared to a simple string matching approach, where each of the samples above would likely have been classified as positive.

CHAPTER IV: RESULTS

4.1 INTRODUCTION

The machine learning-based text classification we performed identified over 1.5 million narrative records as *positive* (i.e. computing technology-related). We mapped each of the positive records to their corresponding event report in the Manufacturer and User Facility Device Experience (MAUDE) database published by the U.S. Food and Drug Administration (FDA). This mapping provided us with additional information the events related to computing technology.

Our analysis of the MAUDE events mostly focused around the dimensions related to our research hypotheses (See Section 1.3). This included: analysis of overall and yearly trends of computing technology-related events; analysis on submitters and reporters of medical device events; and analysis on events associated with serious harm to patients. We also examined if the existing scheme of assigning problem codes to events in the FDA's Medical Device Reporting process was significantly masking problems related to computing technology.

We present the results of these analyses in the sections below.

4.2 EVENTS RELATED TO COMPUTING TECHNOLOGY

The machine learning-based classification of the MAUDE narratives identified 1,531,374 of the 11,689,482 narrative records as computing technology-related. We mapped these positive narrative records to their report records in MAUDE (one report of medical device event can have zero or more narrative records). Using this mapping, we identified a total of 1,155,516 medical device events related to computing technology over the 10-year period (2007 - 2016). Table 22 shows the yearly distribution of medical device events related to computing technology:

Year	**Count**
2007	37,679
2008	24,456
2009	36,687
2010	40,782
2011	87,255
2012	73,237
2013	118,349
2014	203,255
2015	271,409
2016	262,407
Total	**1,155,516**

Table 22: Events Related to Computing Technology

Overall, there was a steady growth in the number of computing technology-related events. Reports of computing technology-related events jumped nearly 7-fold from 37,679 in 2007 to 262,407 in 2016. This growth roughly fits a second order polynomial trend as shown in Figure 17:

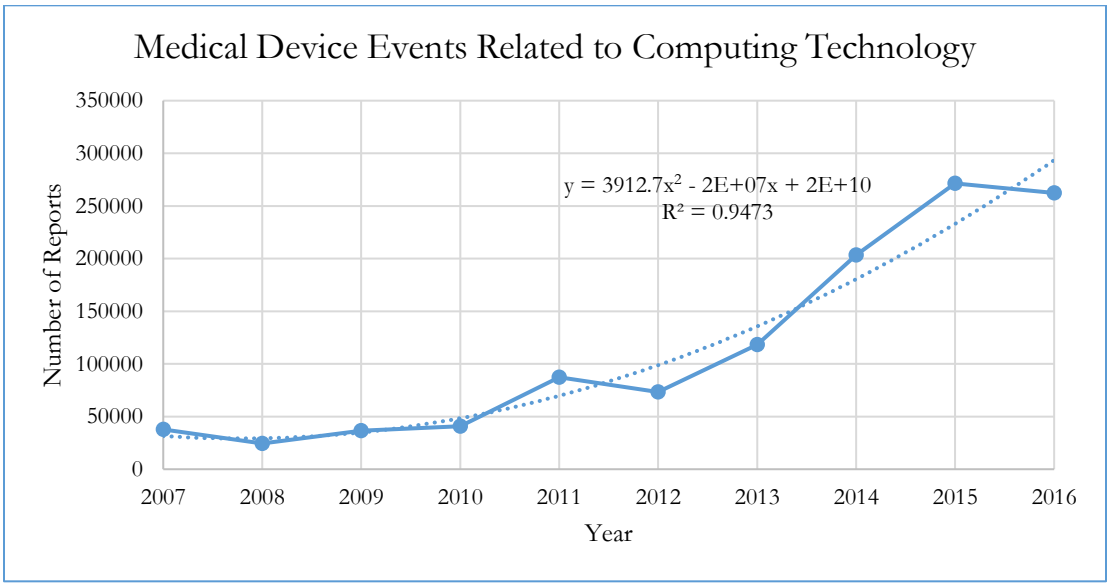


Figure 17: Yearly Growth in Computing Technology Related Events

4.3 Percentage of Total Events

In the 2007 - 2016 period, we found that reports of computing technology-related events accounted for 22.61% of the total 5,110,200 events reported to the FDA. Table 23 shows the yearly breakdown of the portions of computing technology related events:

Year	Total Reports	Computing Technology Related	Percentage
2007	171,322	37,679	21.99%

Year	Total Reports	Computing Technology Related	Percentage
2008	194,424	24,456	12.58%
2009	241,895	36,687	15.17%
2010	303,065	40,782	13.46%
2011	445,118	87,255	19.60%
2012	485,879	73,237	15.07%
2013	679,224	118,349	17.42%
2014	861,826	203,255	23.58%
2015	861,045	271,409	31.52%
2016	866,402	262,407	30.29%
Total	**5,110,200**	**1,155,516**	**22.61%**

Table 23: Percentage of Computing Technology-related Events

Year over year, the percentage of computing technology-related events did not seem to show a clear trend over the 10-year period. However, between 2012 and 2015, percentage of computing technology related events jumped from 15.07% of total reports to 31.52%, an increase of more than 16 percentage points. The rate settled somewhat in 2016 at 30.29%, but that is still the double from 2012, a marked growth.

Also, the year 2007 was rather an anomaly caused by an unusually large number of submissions of device failure events by a single manufacturer. This was explained in Section 1.2.1.2.5. Figure 18 shows the yearly trend of the percent of computing technology-related events.

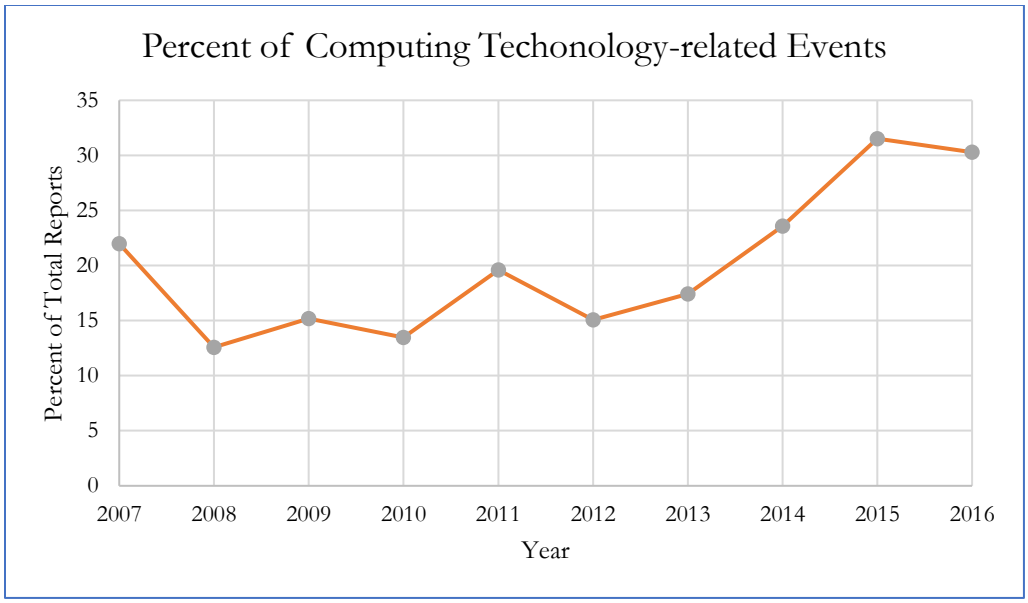


Figure 18: Percent of Computing Technology-related Events

It should be noted that the rise in the percentage of computing technology-related medical device events in recent years should not necessarily be interpreted as a trend of computing technology getting increasingly prone to failures. This is because we do not have information on the prevalence of computer-equipped medical devices year over year. With the rapid proliferation of computing technology across industries, it is conceivable that there are just more medical devices fitted with computing technology on the market every year contributing to the higher number failures.

4.4 Submitters of Computing Technology-related Events

The MAUDE database contains a field called REPORT_SOURCE_CODE on each event record holding one of the following possible values listed in Table 24:

Code	Meaning
P	**Voluntary Report:** Report of a medical device event submitted by the general public.
U	**User Facility Report:** Report of a medical device event submitted by a hospital or a healthcare facility.
D	**Distributor Report:** Report of a medical device event submitted by a distributor of the device.
M	**Manufacturer Report:** Report of a medical device event submitted by the manufacturer of the device.

Table 24: MAUDE Event Report Sources

Classifying reports of medical device events by source shows that for the study period (2007 - 2016) nearly all (99.36%) of the reports of computing technology-related events were submitted by the device manufactures. Table 25 provides the yearly breakdown of computing technology-related events by source:

Year	Manufacturer	Distributor	User Facility	Voluntary	Total
2007	36,694	562	290	133	37,679
2008	23,686	343	299	128	24,456
2009	36,099	80	332	176	36,687
2010	40,095	5	422	260	40,782
2011	86,535	7	461	252	87,255
2012	72,619	15	432	171	73,237
2013	117,751	143	306	149	118,349

Year	Manufacturer	Distributor	User Facility	Voluntary	Total
2014	202,438	239	351	227	203,255
2015	270,571	268	338	232	271,409
2016	261,681	55	368	303	262,407
Total	**1,148,169**	**1,717**	**3,599**	**2,031**	**1,155,516**

Table 25: Submitters of Medical Device Event Reports

4.5 REPORTERS OF COMPUTING TECHNOLOGY-RELATED EVENTS

Note that while manufactures submitted almost all reports of medical device failures related to computing technology, the actual report of the event may have originated from a different source. Starting 2006, the FDA has made the occupation of the event reporter available in the MAUDE dataset. This field (REPORTER_OCCUPATION_CODE) contains one of the following values listed in Table 26:

Code	Description
000	Other
001	Physician
002	Nurse
0HP	Health Professional
0LP	Lay User/Patient
100	Other Health Care Professional
101	Audiologist

Code	Description
102	Dental Hygienist
103	Dietician
104	Emergency Medical Technician
105	Medical Technologist
106	Nuclear Medicine Technologist
107	Occupational Therapist

Code	Description
108	Paramedic
109	Pharmacist
110	Phlebotomist
111	Physical Therapist
112	Physician Assistant
113	Radiologic Technologist
114	Respiratory Therapist
115	Speech Therapist
116	Dentist
300	Other Caregivers
301	Dental Assistant
302	Home Health Aide
303	Medical Assistant
304	Nursing Assistant
305	Patient
306	Patient Family Member or Friend
307	Personal Care Assistant
400	Service and Testing Personnel
401	Biomedical Engineer
402	Hospital Service Technician
403	Medical Equipment Company Technician/Representative
404	Physicist
405	Service Personnel
499	Device Unattended
500	Risk Manager
600	Attorney
999	Unknown
NA	Not Applicable
NI	No Information
UNK	Unknown
*	Invalid Data

Table 26: MAUDE Event Reporter Occupation Codes

The breakdown of the reports of medical device events by reporter's occupation shows an interesting picture. While the largest number of reports came from reporters with

'Other' occupation suggesting a lack of sufficient granularity in FDA's list of occupations, patients were the single largest reporters of medical device events related to computing technology. Table 27 shows the complete list of occupations and their associated number of reports of medical device failure events related to computing technology:

Reporter Occupation	**Number of Reports**
Other	429,324
Patient	375,120
No reporter specified	83,542
Biomedical Engineer	60,151
Not Applicable	45,384
Other Health Care Professional	35,088
Medical Equipment Company Technician/Representative	26,347
Unknown	21,527
Health Professional	19,815
No Information	18,864
Nurse	9,788
Physician	8,873

Reporter Occupation	**Number of Reports**
Patient Family Member or Friend	7,681
Service and Testing Personnel	6,000
Medical Technologist	2,019
Risk Manager	1,923
Respiratory Therapist	1,126
Pharmacist	825
Service Personnel	484
Radiologic Technologist	407
Paramedic	184
Dentist	182
Emergency Medical Technician	152
Other Caregivers	151
Physicist	134

Reporter Occupation	Number of Reports
Hospital Service Technician	131
Physician Assistant	75
Audiologist	57
Medical Assistant	57
Attorney	39
Nursing Assistant	19
Dental Assistant	15
Home Health Aide	9
Dietician	8
Physical Therapist	8
Nuclear Medicine Technologist	6
Speech Therapist	1
Total	**1,155,516**

Table 27: Computing Technology-related Events by Reporter Occupation

As shown in Table 27, when analyzed by the reporter occupation, patients were the original reporters for 375,120 (32.46%) of the computing technology-related medical device events. However, as discussed in Section 4.4, voluntary submissions constituted a negligible portion (2,031 or 0.18%) of the total reports. What this tells us is that even though patients are the ones who discover computing technology-related medical device issues in nearly a third of the total cases, they instead report the issues to the manufactures or distributors of the suspect devices, rather than to the FDA.

4.6 PUBLIC-REPORTED MEDICAL DEVICE EVENTS

For this analysis, we defined 'public' as reporters that were not healthcare professionals or associated with manufacturers and distributors. To avoid a potentially

incorrect extrapolation, we limited the definition to the following occupations from the list of occupations in the MAUDE database: Lay User/Patient (Code: 0LP), Patient (305) and Patient Family Member or Friend (306) essentially restricting the events to those reported by the patients or their representatives. We then compared the proportion of public-reported events and the proportion of public-reported events related to computing technology. Table 28 shows the result of this comparison for each of the study years:

Year	Total Events	Public-Reported	Percent		Computing Technology-related Events	Public-Reported	Percent
2007	171,322	14,814	8.65%		37,679	1,279	3.39%
2008	194,424	15,459	7.95%		24,456	2,326	9.51%
2009	241,895	14,785	6.11%		36,687	6,544	17.84%
2010	303,065	15,138	4.99%		40,782	6,819	16.72%
2011	445,118	19,901	4.47%		87,255	5,515	6.32%
2012	485,879	41,340	8.51%		73,237	14,521	19.83%
2013	679,224	63,072	9.29%		118,349	31,419	26.55%
2014	861,826	132,538	15.38%		203,255	77,409	38.08%
2015	861,045	182,621	21.21%		271,409	123,140	45.37%
2016	866,402	191,067	22.05%		262,407	113,829	43.38%
Total	**5,110,200**	**690,735**	**13.52%**		**1155,516**	**382,801**	**33.13%**

Table 28: Public-reported Medical Device Events

Over the ten study years (2007 – 2016), based on the reporter occupation code, the general public reported a higher percentage (33.13%) of medical device events related to computing technology than they did overall events (13.52%). While the trend shows that the general public is increasingly reporting medical device events (both computing technology-related and overall), the growth is remarkable for computing technology-related events in the more recent years. For example, in 2016, the general public reported 43.38% of the computing technology-related events. We should re-iterate that the "reporter" in this sense is not the *submitter* of the event reports to the FDA, which in almost all cases are the manufacturers of medical devices. Instead, reporter in this sense is the individual who discovers the problem in a device that ultimately results in a submission (likely by the manufacture of the suspect device) to the FDA.

It is also important to refrain from concluding that the general public is increasingly more engaged in reporting medical device events based on this data. This trend may instead be because there are just more computing technology-equipped medical devices (e.g. glucose meters, insulin pumps, cardioverter defibrillators, etc.) in the consumers' hands. Figure 19 shows the trend of public-reported events over the study period:

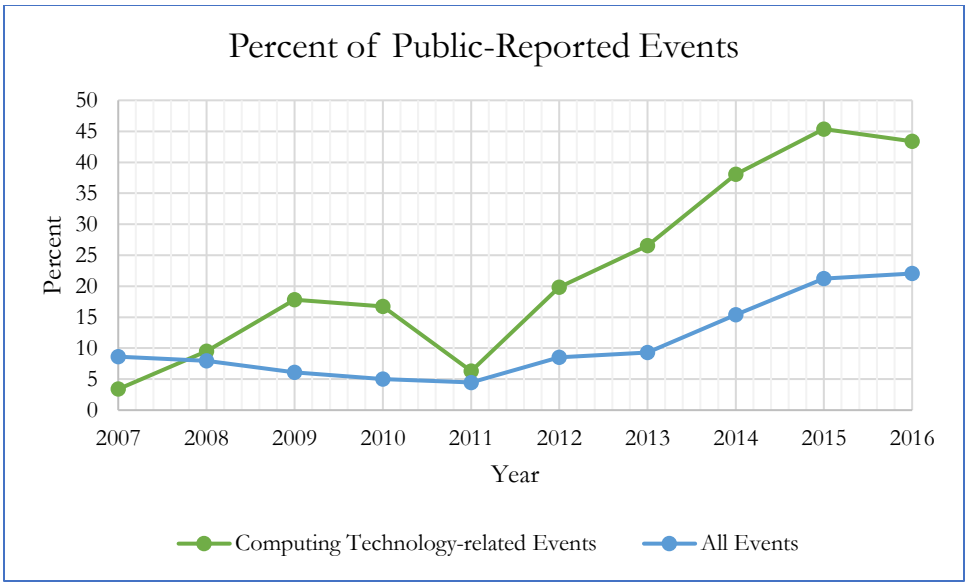


Figure 19: Percent of Public-Reported Events

4.7 Patient Deaths Associated with Computing Technology-related Events

The MAUDE database contains multiple fields that indicate the consequence of a medical device event. First, the event master table contains a field called 'EVENT_TYPE', which holds one of the following values:

Event Type Code	**Meaning**
D	Death
IJ	Injury
IL	Injury
IN	Injury
M	Malfunction

Event Type Code	Meaning
O	Other
*	No answer provided

Table 29: MAUDE Event Type Codes

There is no explanation provided on why there are three different codes for injury. Per the FDA, the EVENT_TYPE field is relevant for reports submitted by manufactures only.

The MAUDE database also includes a patient information table that contains applicable (if any) treatment and outcome information associated with each reported event. This SEQUENCE_NUMBER_OUTCOME field contains a semi-colon (;) separated pair of patient sequence number and outcome code (Example: "1. L; 2. R; 3. H"). The patient outcome code has one of the values listed in Table 30:

Outcome Code	Meaning
A	Not Applicable
C	Congenital Anomaly
D	Death
H	Hospitalization
I	No Information
L	Life Threatening
O	Other
R	Required Intervention
S	Disability
U	Unknown

Outcome Code	Meaning
*	Invalid Data

Table 30: MAUDE Patient Outcome Codes

When queried independently, there were a total of 3,313 records found in the event master table with the Event Type of 'D' (i.e. death) and related to computing technology. On the other hand, the patient information table contained 2,188 records with the outcome data involving the outcome code of 'D' (i.e. death).

While most events with the Event Type of death also contained an associated patient outcome record indicating death, there were several instances where data was in conflict. For our analysis, we selected *unique* events that either had the Event Type of death or contained death as the patient outcome. This yielded a total of 3,449 unique computing technology-related events associated with patient death over the 10-year period. This was 3.33% of all 103,372 medical device events associated with patient death for the same period.

Table 31 shows the annual breakdown of the number of computing technology-related events associated with patient death:

Year	Number of Death Events (All)	Number of Death Events (Computing Technology)
2007	3,177	144
2008	4,417	173
2009	5,352	279
2010	6,557	305

Year	Number of Death Events (All)	Number of Death Events (Computing Technology)
2011	10,220	229
2012	10,635	255
2013	11,225	333
2014	25,284	419
2015	17,609	604
2016	8,896	708
Total	**103,372**	**3,449**

Table 31: Computing Technology-related Events Associated with Patient Death

The yearly breakdown of the number of computing technology-related events associated with patient death shows that except for two years (2011, 2012), events associated with patient death have been rising every year over the 10 years (2007 - 2016). From 144 in 2007 to 708 in 2016, annual number of computing technology-related events associated with patient death increased nearly five-fold in the 10 years. Figure 20 illustrates this trend:

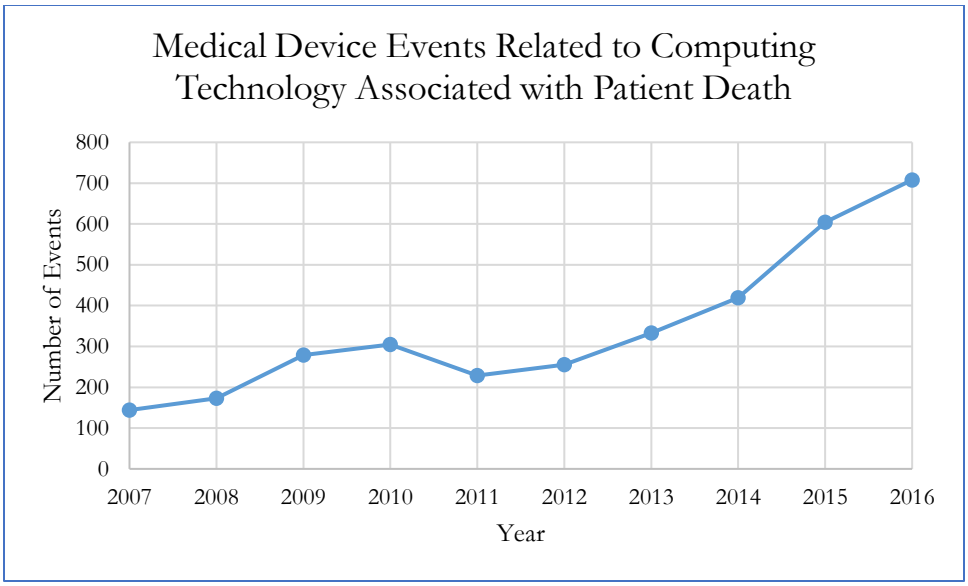


Figure 20: Computing Technology-related Events Associated with Patient Death

This increasing trend, however, does not necessarily mean that the computing technology in medical devices is getting more fatal. This is because, as a percentage of total computing technology-related events, events associated with patient death have remained largely flat (less than 1% of all computing technology-related events) over the 10 years, as shown in Figure 21:

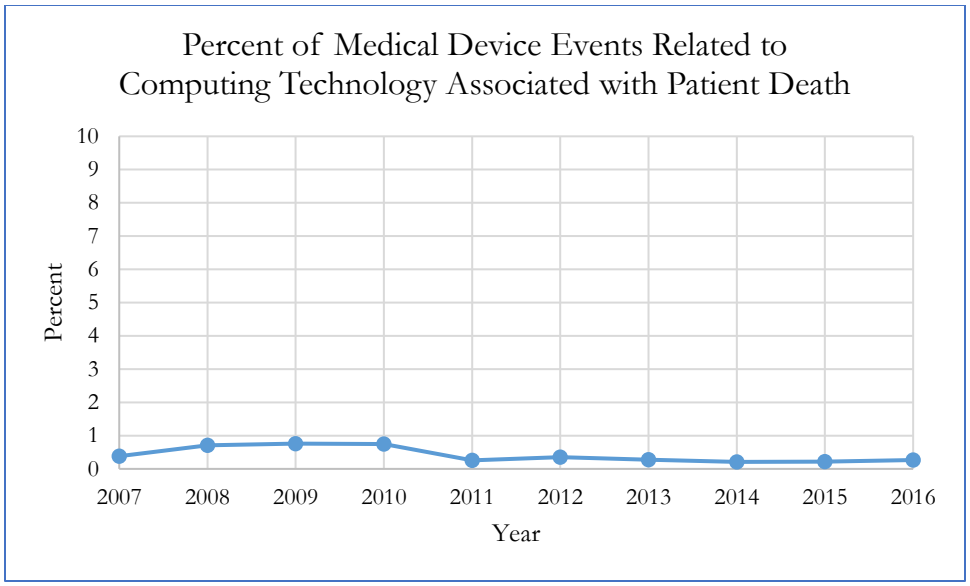


Figure 21: Percent of Medical Device Events Associated with Patient Death

4.8 Patient Injuries Associated with Computing Technology-related Events

To explore potential patient injuries associated with computing technology-related medical device events, we extended our analysis to determine from the computing technology-related events a subset associated with an injury. More specifically, we filtered all reports of computing technology-related medical device failures in the MAUDE database by Event Type (IJ, IL or IN) and Patient Outcome (H, L or S for hospitalization, life-threatening and disability respectively). Overall, there were a total of 104,754 computing technology-related medical device events associated with serious patient injuries in the 10-year period (2007 - 2016). This translates to 9.07% of all computing technology-related events. Table 32 shows the annual number of injury events and their percentage of the total computing technology-related events:

Year	Injury Events	Percent of Computing Technology-related Events
2007	3,118	8.28%
2008	3,755	15.35%
2009	4,061	11.07%
2010	5,052	12.39%
2011	5,552	6.36%
2012	6,794	9.28%
2013	7,893	6.67%
2014	12,228	6.02%
2015	22,263	8.2%
2016	34,038	12.97%
Total	**104,754**	**9.07%**

Table 32: Computing Technology-related Events Associated with Patient Injury

We found that the number of computing technology-related events associated with patient injury grew every year in the study period (2007 - 2016). This growth was particularly steep since 2013. In 2013, the number events associated with patient injury were 7,893, whereas 2016 saw an over four-fold increase to 34,038. Figure 22 illustrates the annual growth in the number of computing technology-related events associated with patient injury:

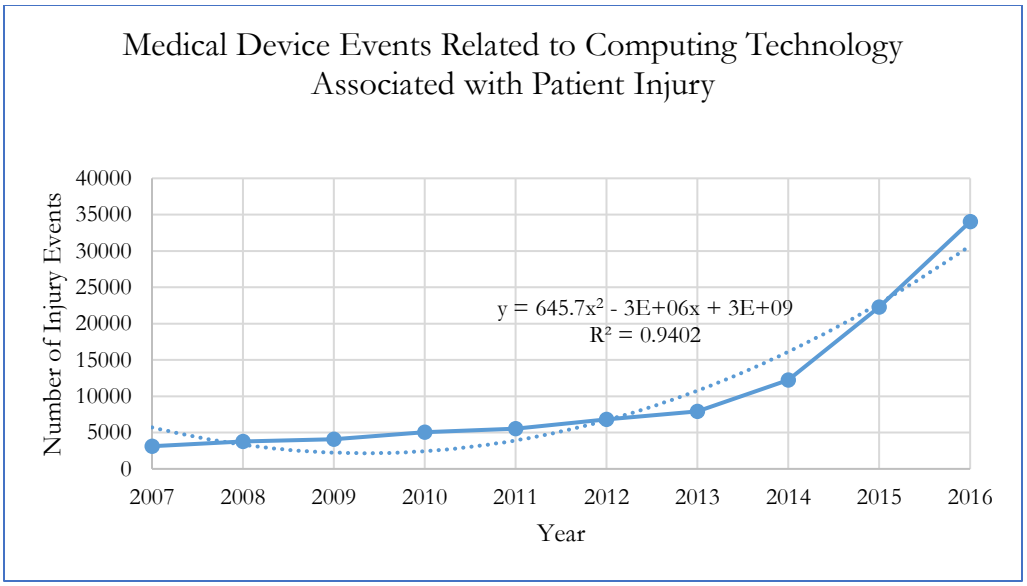


Figure 22: Computing Technology-related Events Associated with Patient Injury

While there has been a significant growth in the number of computing technology-related events associated with patient injury reported over the 10 study years on an absolute basis, this does not necessarily suggest computing technology getting more dangerous. This is because as a percentage of all computing technology-related events, events associated with patient injury do not show a similarly consistent overall trend, as shown in Figure 23:

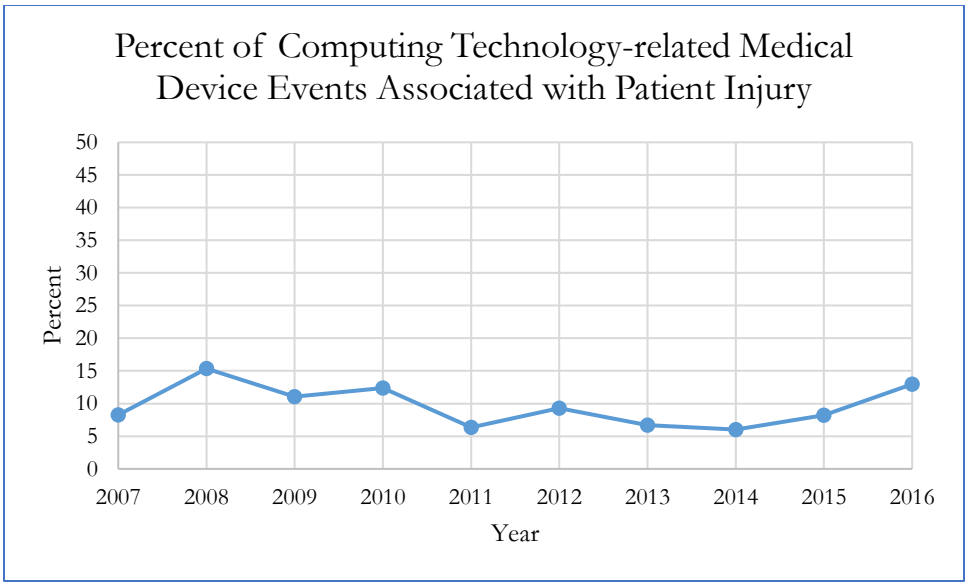


Figure 23: Percent of Computing Technology-related Medical Device Events Associated with Patient Injury

4.9 MASKING OF COMPUTING TECHNOLOGY-RELATED PROBLEMS

In this analysis, we wanted to explore if, in fact, the FDA's current scheme of assigning problem codes to an event in its Medical Device Reporting program is masking computing technology-related problems (we discussed the problem in the results of our preliminary research in Section 1.2.1.2). More specifically, we wanted to examine if the machine learning-based classification approach we followed had uncovered computing technology-related events that were misclassified in the MAUDE datasets published by the FDA.

Although there is a fundamental difference between the goal of our machine learning research (to identify if an event is computing technology *related*) and the problem code assignment in MAUDE (to *describe* the observed device problem), we started with a simple comparison between the number of computing technology-related events identified through

the current problem code assignment scheme in MAUDE and through the machine learning-based approach we implemented in this research. This comparison is presented in Table 33:

Year	All Events	Problem Codes in MAUDE database		Machine Learning-based Classification	
		Count	Percent of All	Count	Percent of All
2007	171,322	9,766	5.7%	37,679	21.99%
2008	194,424	907	0.47%	24,456	12.58%
2009	241,895	740	0.31%	36,687	15.17%
2010	303,065	431	0.14%	40,782	13.46%
2011	445,118	878	0.2%	87,255	19.6%
2012	485,879	548	0.11%	73,237	15.07%
2013	679,224	276	0.04%	118,349	17.42%
2014	861,826	676	0.08%	203,255	23.58%
2015	861,045	2,721	0.32%	271,409	31.52%
2016	866,402	3,301	0.38%	262,407	30.29%
Total	**5,110,200**	**20,244**	**0.4%**	**1,155,516**	**22.61%**

Table 33: Comparison of Problem Codes in MAUDE and Machine Learning-based Relational Classification

As expected, the comparison showed a dramatic difference between the problem code assignment in MAUDE based on the FDA's Device Problem Code Hierarchy (See Section 0 for more information), and the machine learning-based classification we applied in

our research. However, considering the different scopes (*functional description* in the case of MAUDE classification vs. *relational* in our classification), the difference could not entirely be viewed as the set of misclassified events in MAUDE.

We were, nevertheless, intrigued by the extremely low rate of problem code assignment of computing technology-related problems in the MAUDE database. To investigate this further, we implemented a method which we will discuss in the section below.

4.9.1 Sampling of Events with Strong Computing Technology Causality

To assess the effectiveness of the problem code assignment scheme in the MAUDE database, we needed to identify events that were truly *caused* by computing technology and manifested to the User as such. To do this, we took the randomized, positive seed data (See Section 3.2.4.2.1) from our machine learning experiment and searched for the presence of 55 very specific phrases that would indicate the causality. The specific keywords used in our search are listed in Table 34 below:

COMPUTER ANOMALY	COMPUTER FREEZE	CORRUPT HARD DRIVE
COMPUTER CRASH	COMPUTER FROZE	CORRUPT SOFTWARE
COMPUTER DEFECT	COMPUTER PROBLEM	DATABASE ANOMALY
COMPUTER ERROR	CORRUPT DATABASE	DATABASE CORRUPT
COMPUTER FAIL		

DATABASE DEFECT
DATABASE ERROR
DATABASE FAIL
DATABASE PROBLEM
DISPLAY FREEZE
DISPLAY FROZE
FILE CORRUPT
HARD DISK FAIL
HARD DRIVE CORRUPT
HARD DRIVE ERROR
HARD DRIVE FAIL
MEMORY ERROR
MOTHER BOARD ERROR
MOTHER BOARD FAIL
MOTHERBOARD ERROR

MOTHERBOARD FAIL
NETWORK DEFECT
NETWORK ERROR
NETWORK ISSUE
NETWORK PROBLEM
SCREEN FREEZE
SCREEN FROZE
SOFTWARE ANOMALY
SOFTWARE BUG
SOFTWARE CORRUPT
SOFTWARE CRASH
SOFTWARE DEFECT
SOFTWARE ERROR
SOFTWARE FAIL
SOFTWARE FREEZE
SOFTWARE FROZE

SOFTWARE MALFUNCTION
SOFTWARE PROBLEM
SOFTWARE UPDATE FAIL
SOFTWARE UPGRADE FAIL
SYSTEM FILE ERROR
USER INTERFACE CRASH
USER INTERFACE DEFECT
USER INTERFACE ERROR
USER INTERFACE FAIL
USER INTERFACE FREEZE
USER INTERFACE FROZE

Table 34: Computing Technology Phrases for Causality Sampling

The search for records in the positive seed dataset with an exact match for the specific causality phrases yielded 102 narrative records. Identifiers for all of these records are listed in Appendix H.

4.9.2 Analysis

For each of the 102 sample records, we analyzed how the MAUDE dataset classified them in terms of the problem codes. Table 35 shows the summary of the state of the problem code assignment in MAUDE:

Classification State	**Count**
Correctly classified as computing technology related	11
No code specified	64
Different code specified	27
Total	**102**

Table 35: Accuracy of Problem Code Assignment in MAUDE Database

The results showed that out of 102 records, a vast majority (89.21%) of medical device events in our sample set *caused* by computing technology were either not classified with any problem code; or misclassified in the MAUDE database. Our machine learning-based classification, on the other hand, accurately classified 92 (90.19%) of the 102 records and counted each of the remaining 10 as a potential false negative (See Section 3.4.3).

Using the narratives as the basis for classification, we present in Table 36 below 20 example records from our sample set that either contain no problem code or contain codes not related to computing technology in the MAUDE database published by the FDA:

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
1214830	957549	DURING AN INVESTIGATION OF AN INCIDENT REPORTED BY ONE OF INTELERAD'S FIELD PERSONNEL IN (B)(4), INTELERAD MEDICAL **IDENTIFIED A SOFTWARE DEFECT IN THE SYSTEM** THAT WOULD IN CERTAIN CIRCUMSTANCES PREVENT CERTAIN IMAGES FROM BEING DISPLAYED WITH NO WARNING OR ALARM TO NOTIFY THE USER OF A FAILURE. WHEN RETRIEVING IMAGES USING THE DICOM SERVICE, AND WHEN THESE IMAGES ARE PROCESSED IN A BATCH, **IT IS POSSIBLE THAT THE INTELEVIEWER DATABASE ON THE WORKSTATION MAY NOT BE UPDATED AND SOME IMAGES MAY NOT BE RETRIEVED.** DEPENDING ON HOW THE IMAGES ARE LOADED, THIS MAY RESULT IN EITHER THE IMAGES NOT BEING SHOWN AT ALL (I.E. INTELEVIEWER REPORTING THAT THERE ARE FEWER IMAGES IN A SERIES THAN WERE SENT BY THE SERVER), OR A DELAY IN SHOWING THE IMAGES BECAUSE THEY HAVE TO BE LOADED AGAIN. BASED ON A REVIEW OF THE POSSIBLE HEALTH HAZARDS WITH A LOCAL CUSTOMER, THERE WAS A CONCLUSION THAT SINCE THE PROBLEM COULD NOT ALWAYS BE DETECTED, THERE WAS A RISK OF A POSSIBLE MISDIAGNOSIS. THE INTELEVIEWER MODULE IS IN BOTH THE INTELEVIEWER WORKSTATION AND THE INTELEPACS PRODUCTS.	No code assigned
1220838	957998	REPORTEDLY, DURING USE ON A PATIENT, THE VENTILATOR **DISPLAY FROZE WITH THE FOLLOWING ERROR MESSAGE: "CAN NOT OPEN BITMAP FILES. ALL.C/360"**. THERE WAS NO AUDIBLE ALARM; ONLY ALARM LAMP WAS LIT. PLEASE NOTE	1019 - Not audible alarm

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		THAT THERE WAS NO PT INJURY OR MEDICAL INTERVENTION OCCURRED IN THIS CASE. INVESTIGATION OF THIS ISSUE HAS FOUND THAT WHEN THIS ERROR MESSAGE OCCURS THE PT CONTINUES TO BE VENTILATED, HOWEVER, IF A SUBSEQUENT ALARM EVENT WERE TO OCCUR, THERE WOULD BE A VISUAL ALARM, BUT NO AUDIBLE ALARM WOULD SOUND TO ALERT THE CAREGIVER OR HOSPITAL STAFF. THIS SPECIFIC ERROR MESSAGE HAS ONLY BEEN **NOTED IN THE FOREIGN LANGUAGE 3.1B VERSION OF THE VENTILATOR SOFTWARE.**	
1455879	19206289	IT WAS REPORTED THAT A SITE'S **HANDHELD COMPUTER KEPT HAVING AN "SQL" ERROR**. THE COMPANY REP PERFORMED TROUBLESHOOTING, BUT THE EVENT WAS NOT RESOLVED. THE HANDHELD WAS SENT TO THE MFR FOR ANALYSIS AND THE SITE RECEIVED A NEW HANDHELD COMPUTER. THE HANDHELD COMPUTER AND SOFTWARE WERE RECEIVED BY THE MFR FOR ANALYSIS. PRODUCT ANALYSIS WAS PERFORMED ON THE HANDHELD COMPUTER, AND THERE WERE NO ANOMALIES NOTED. **ANALYSIS OF THE SOFTWARE INDICATED THAT THERE WAS A CORRUPTION OF THE DATABASE**. THE CAUSE FOR THE CORRUPT DATABASE COULD NOT BE DETERMINED.	No code assigned
1571353	8425432	(B) (4). THIS EVENT CONCERNS A DEVICE THAT WAS MANUFACTURED AND USED OUTSIDE THE UNITED STATES. IN RESPONSE TO THE WARNING LETTER THAT THE UNITED STATES FOOD AND DRUG ADMINISTRATION ISSUED TO ELA	1440 - Pacer found in back-up mode

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		MEDICAL (DATED 11/06/2009), ELA IS FILING THIS MDR AT THIS TIME. (B) (4). CONCLUSION: THE OBSERVED RESET RESULTED FROM AN **ALTERATION OF ONE BYTE OF THE DOWNLOADED SOFTWARE**; THE ROOT CAUSE OF THIS ALTERATION COULD NOT BE IDENTIFIED WITH CERTAINTY BUT THE **MOST PROBABLE HYPOTHESIS IS A TRANSIENT SOFTWARE ERROR** ("SINGLE EVENT UPSET" OR "BIT-FLIP") DUE TO TRANSFER OF ENERGY FROM A CHARGED PARTICLE. THIS IS A KNOWN, RARE, AND NON DESTRUCTIVE PHENOMENON IN MICROELECTRONIC CIRCUITS. THE LOG FILE OF THE INTERROGATION DURING THE INCIDENT WAS NOT AVAILABLE; THEREFORE, IT WAS NOT POSSIBLE TO EXPLAIN IF THE LONG TIME TO STOP THE INTERROGATION WAS CAUSED BY SOME POSSIBLE TELEMETRY ERRORS OR IF IT IS MORE LIKELY DUE TO THE TIME NECESSARY TO SAVE ALL THE EXPERT FILE. FINALLY THE RESET WAS FORCED BY THE PROGRAMMER WITH A NORMAL RE-INITIALIZATION OF THE DEVICE, NO FURTHER ACTIONS ARE NECESSARY. THE DEVICE HAS BEEN RECYCLED AT THE FINAL STEP OF THE MANUFACTURING PROCESS IN ORDER TO RELOAD THE SOFTWARE AND THE PARAMETERS (THE WARNING FOR MICROPROCESSOR RESET HAS BEEN CONSEQUENTLY ERASED). IT WILL BE RELEASED BACK FOR DISTRIBUTION AFTER CONTROL OF THE OPERATIONS PERFORMED IN COMPLIANCE WITH OUR MANUFACTURING AND QUALITY PROCEDURE.	

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
1819444	16412636	THE USER RECEIVED ANALYZER ALARMS AND A QUESTIONABLE GLUCOSE RESULT FOR ONE PATIENT SAMPLE FROM THE COBAS C111 ANALYZER. THE INITIAL RESULT WAS 9.4 MG/DL. THE USER SHUT DOWN AND REBOOTED THE ANALYZER, THEN REPEATED THE PATIENT SAMPLE. THE REPEAT GLUCOSE RESULT WAS 187.4 MG/DL TWICE AND WAS REPORTED. THE INITIAL RESULT WAS NOT REPORTED OUTSIDE THE LABORATORY AND THE PATIENT WAS NOT AFFECTED. THE GLUCOSE REAGENT LOT NUMBER WAS 62255801. THE FIELD SERVICE REPRESENTATIVE **DETERMINED THERE WAS A SOFTWARE ERROR AND REINSTALLED THE SYSTEM SOFTWARE**. HE VERIFIED THE ANALYZER OPERATION BY PERFORMING SERVICE DIAGNOSTICS. THE USER RAN CALIBRATION AND QUALITY CONTROL WITH ACCEPTABLE RESULTS.	2458 - Low test results
2275187	18202714	(B)(4). EVALUATION SUMMARY: THE REPORTED CONDITION OF A COLLEAGUE PUMP WITH A FAILURE 12:206:1604 WAS CONFIRMED AND NOT REPRODUCED DURING PRODUCT EVALUATION. THE ROOT CAUSE OF THIS CONDITION WAS ASSIGNED TO **INVALID VALUE IN THE RAM (RANDOM ACCESS MEMORY), WHICH IS A SOFTWARE FAILURE.** THE PUMP WAS POWERED OFF AND ON AND THE FAILURE DID NOT RECUR. SOFTWARE ISSUES ARE NOT ASSOCIATED WITH HARDWARE MALFUNCTIONS. THEREFORE THERE WILL BE NO REPAIRS MADE TO CORRECT THE REPORTED PROBLEM. A SERVICE HISTORY REVIEW REVEALED THAT THIS DEVICE HAS	No code assigned

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		NOT BEEN SERVICED PRIOR TO THIS EVENT. A DEVICE HISTORY RECORD REVIEW WAS PERFORMED FINDING NO EXCEPTION, NONCONFORMANCE, OR REWORK THAT OCCURRED DURING THE MANUFACTURING OF THE COMPLAINT LOT OR SERIAL NUMBER.	
2359547	9898715	AFTER FURTHER INVESTIGATION BY THE QUALITY ENGINEERS, THE **ROOT CAUSE OF THE REPORTED CONDITION WAS ASSIGNED TO SOFTWARE FAILURE**. SOFTWARE ISSUES ARE NOT ASSOCIATED WITH HARDWARE MALFUNCTIONS. THEREFORE THERE WERE NO REPAIRS MADE TO CORRECT THE REPORTED PROBLEM. A SERVICE HISTORY REVIEW REVEALED THAT THIS DEVICE WAS PREVIOUSLY SERVICED FOR FAILURES/PROBLEMS THAT WERE SAME AS OR SIMILAR TO THE CURRENT PROBLEM.	No code assigned
2449480	2482860	THE CUSTOMER REPORTED THE SYSTEM DISPLAYED A **FILE CORRUPT ERROR MESSAGE AND WOULD NOT BOOT UP**. THERE IS NO REPORT OF PATIENT INJURY.	No code assigned
2506612	21921360	INVESTIGATION OF THE RETURNED PRODUCT **DETERMINED THE CAUSE TO BE ISOLATED TO SOFTWARE CORRUPTION**. ALTHOUGH GLUCOSE RESULTS MAY BE DELAYED, BLOOD GLUCOSE COULD BE DETERMINED BY ALTERNATE MEANS, INCLUDING USE OF ANOTHER BLOOD GLUCOSE METER, SEEING A PHYSICIAN (AS RECOMMENDED IN PRODUCT LABELING), OR BY SEEKING TREATMENT AT A HEALTH CARE FACILITY. (B)(4).	No code assigned

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
3320545	27443146	(B)(4) THIS WAS A PULMONARY VEIN ISOLATION (PVI) ABLATION FOR AN ATRIAL FIBRILLATION (AFIB) PROCEDURE. THE SYSTEM WAS VERY SLOW WHEN MOVING FROM ONE MAP TO ANOTHER OR WHEN OPENING A NEW MAP/REMAP. THE SYSTEM WOULD FREEZE. THE WAIT WAS SOMETIMES 10-15 MINUTES. WHEN MANIPULATING THE MAP, IT DID NOT MOVE SMOOTHLY, BUT JUMPY. ON EXAMINATION OF THE PATIENT THE NEXT MORNING, IT WAS DISCOVERED THAT THE PATIENT HAD DEVELOPED A SLIGHT PERICARDIAL EFFUSION. IT DID NOT REQUIRE ASPIRATION AND THE PATIENT WAS DISCHARGED LATER IN THE DAY. THE DEFECTIVE WORKSTATION WAS REPLACED WITH ANOTHER ONE WHICH WAS DELIVERED TO THE ACCOUNT. THEN THE DEFECTIVE WORKSTATION WAS SENT TO THE HAIFA TECHNOLOGY CENTER (HTC) FOR INVESTIGATION. HTC FOUND THAT THE **CORRUPT SOFTWARE DATABASE CAUSED THE ISSUE.** THE REASON COULD NOT BE IDENTIFIED. THE DEVICE HISTORY RECORD (DHR) REVIEW WAS PERFORMED. NO ANOMALIES WERE NOTED IN MANUFACTURING OR SERVICE.	No code assigned
3448598	21490240	THE CUSTOMER CONTACT REPORTED A WHITESCREEN ERROR. THE DEVICE WAS RETURNED TO THE BIOMEDICAL DEPARTMENT FROM THE 2L MEDICAL UNIT OR 2L SURGICAL UNIT WITH A **REPORT OF SOFTWARE ERROR**. NO SPECIFIC PATIENT INFORMATION, PUMP PROGRAMMING, OR EVENT DETAILS WERE NOT AVAILABLE. THERE WERE NO REPORTS OF ANY ADVERSE PATIENT EVENTS AND NO REPORTED	405 - Alarm, Audible 1663 - Device inoperable

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		DELAYS OF CRITICAL THERAPIES WHILE THE DEVICE WAS IN CLINICAL USE. DURING REVIEW OF THE DEVICE HISTORY, **AN UNSPECIFIED WHITESCREEN ERROR WAS NOTED**. THE CUSTOMER CONTACT INDICATED THAT THE DEVICE WAS RESET AND THEN THE DEVICE WAS RETURNED TO CLINICAL SERVICE. NO SPECIFIC DETAILS WERE PROVIDED. THOUGH REQUESTED, NO ADDITIONAL INFORMATION WAS PROVIDED.	
3795936	15550554	FINAL ANALYSIS FOUND THE DEVICE EXHIBITED AND RADIO FREQUENCY TELEMETRY ANOMALY **DUE TO A SOFTWARE ANOMALY**.	No code assigned
3995715	12441074	ADDITIONAL TESTING WAS CONDUCTED ON THE RETURNED COMPUTER. THE **COMPUTER HARD DRIVE FAILED** A PROFESSIONAL HARDWARE DIAGNOSTICS (PHD) TEST. THE SMART DATA ANALYSIS SHOWED BAD SECTORS. **THE HARD DRIVE WAS FOUND DEFECTIVE AND TO HAVE CAUSED THE OPERATING SYSTEM ISSUE.** AS PREVIOUSLY REPORTED, THE COMPUTER WAS REPLACED TO RESOLVE THE ISSUE.	No code assigned
4250870	16439876	CRACKED RESERVOIR TUBE LIP, CRACKED BATTERY TUBE THREADS, MINOR SCRATCHES ON DISPLAY WINDOW AND CRACKED CASE NEAR DISPLAY WINDOW CORNERS NOTED DURING VISUAL INSPECTION. PUMP PASSED FUNCTIONAL TEST INCLUDING PRIME/A33, DISPLACEMENT, BASIC OCCLUSION, OCCLUSION AND EXCESSIVE NO DELIVERY ALARM TESTS. NO SIGNAL TOO LOW ALARMS NOTED DURING TESTING. SPANISH SOFTWARE ANOMALY ALARMS NOTED IN ALARM HISTORY **DUE TO CORRUPTED HISTORY FILES.** THE	No code assigned

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		INSULIN PUMP INVOLVED IN THIS EVENT IS THE PARADIGM REAL-TIME VEO INSULIN INFUSION PUMP, WHICH IS NOT MARKETED IN THE UNITED STATES. HOWEVER, THE DEVICE IS SIMILAR TO THE PARADIGM REAL-TIME INSULIN INFUSION PUMP, WHICH IS MARKETED IN THE UNITED STATES. THIS MDR RELATED TO THE PUERTO RICO MANUFACTURING SITE HAS BEEN ASSIGNED A MEDWATCH NUMBER FROM THE MEDTRONIC MINIMED (B)(4).	
4317222	22038450	**SOFTWARE ERROR** - INCORRECT DISPLAY OF THE GRID ADDRESS OF THE BRACHYTHERAPY TEMPLATE WHEN USING BI-PLANE TRANS-RECTAL PROBE C41L47RP WITH THE ARIETTA 60 & 70 DIAGNOSTIC ULTRASOUND SYSTEM. **THE GRID ADDRESS (A THROUGH M) IS INVERTED HORIZONTALLY, DUE TO A SOFTWARE ERROR**. THE PHENOMENON OCCURS WHEN CONNECTING PROBE MODEL# C41647RP AND BRACHYTHERAPY IS ACTIVATED. RISK ASSESSMENT OF HEALTH HAZARD: DURING THERAPEUTIC PLANNING, THERE IS POTENTIAL OF INSERTION OF RADIATION SEEDS IN AN INCORRECT LOCATION WITHIN BODY. SOLUTION: UPDATE THE SOFTWARE WITH SERVICE PACK SP-AR-60-S123-USA AND SP-AR-70-S123-USA. THESE TWO SERVICE PACKS WILL UPDATE SOFTWARE AND CORRECT THE ISSUE. INVESTIGATION: REVIEW OF ALL AFFECTED SERIAL NUMBERS SOLD WITHIN THE US FOR SERVICE AND COMPLAINT DID NOT IDENTIFY ANY CUSTOMER COMPLAINT OR ISSUE AT THIS TIME. REVIEW OF ALL SALES IDENTIFIED ONE CUSTOMER WITH THE	No code assigned

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		ARIETTA 70 SYSTEM AND C41L47RP PROBE COMBINATION. NOTIFICATION WAS IMMEDIATELY SENT AND INVESTIGATION INDICATED THAT NO PATIENT WAS AFFECTED BY ISSUE. TO DATE, THE FEATURE COMBINATION HAS NOT BEEN UTILIZED.	
4992244	22710293	IT WAS REPORTED THAT THE SYSTEM **HAD A HARD DRIVE ERROR** AND WOULD NOT BOOT UP. THERE WAS NO PATIENT INJURY OR DEATH REPORTED.	1476 - Failure to power-up
5087987	26177195	D|1||IN THIS EVENT IT WAS REPORTED THAT A CUSTOMER WAS ABLE TO USE A GUIDE ONLY FOR THE RIGHT SIDE. THE DOCTOR WAS NOT ABLE TO PERFORM THE SURGERY ON THE LEFT SIDE BECAUSE THE ATLANTIS ABUTMENT WAS ROTATED. **INVESTIGATION SHOWS THAT A SOFTWARE PROBLEM IN THE LIBRARY LEAD TO THE ROTATED ABUTMENT POSITION**. THE DENTIST WANTED TO DO IMMEDIATE LOADING IN A SHOW CASE, SO HE DECIDED NOT TO PLACE THE IMPLANT.	2588 - Defective item
5481181	42058548	THE INITIAL MDR 1226181-2016-00108 WAS FILED ON MARCH 4, 2016. ADDITIONAL INFORMATION (03/21/2016): SIEMENS HEALTHCARE DIAGNOSTICS HAS **CONFIRMED A SOFTWARE DEFECT** WHICH, IN A VERY SPECIFIC SET OF CIRCUMSTANCES, RESULTS IN THE DIMENSION VISTA SYSTEM OMITTING AN ALIQUOT PROBE RINSE BETWEEN SAMPLE ASPIRATIONS WHEN PROCESSING TUBES IN SAMPLE RACKS THAT ARE FRONT LOADED ON THE DIMENSION VISTA SYSTEM. URGENT MEDICAL DEVICE CORRECTION (UMDC) VSW16-01.A.US WAS	1384 - Mechanical issue 2914 - Device Operational Issue

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		SENT TO CUSTOMERS IN THE UNITED STATES IN MARCH 2016 AND CORRESPONDING URGENT FIELD SAFETY NOTICE (UFSN #VSW16-01.A.OUS) TO ALL OUTSIDE US DIMENSION VISTA CUSTOMERS. THE UMDC AND UFSN ARE ENTITLED "DIMENSION VISTA SYSTEM MAY NOT PERFORM ALIQUOT PROBE RINSE." THE UMDC AND UFSN DESCRIBE THE SOFTWARE DEFECT, WHAT SAMPLES ARE NOT IMPACTED, THE RISK TO HEALTH, AND ACTIONS TO BE TAKEN BY THE CUSTOMER TO MINIMIZE OR ELIMINATE THE IMPACT OF THE SOFTWARE DEFECT.	
5690889	48656426	DEVICE EVALUATION: THE DEVICE HAS BEEN RETURNED AND EVALUATED BY PRODUCT ANALYSIS ON 06/10/2016 WITH THE FOLLOWING FINDINGS: UPON PUMP POWER UP, A CALL SERVICE ALARM 069 WAS REPRODUCED AND WAS UNABLE TO BE CLEARED POST-REBOOT ATTEMPT. THE CALL SERVICE ALARM 069 FAILURE WAS DEFINED AS A **LANGUAGE FILE CORRUPTION UPON THE LANGUAGE TEXT RETRIEVAL AND WAS RECORDED IN THE BLACK BOX DATA** AS A CALL SERVICE ALARM 87 FAILURE. THE INVESTIGATORS DETERMINED THAT THE ERROR IS RESIDENT TO U39 FLASH CHIP FAILURE. UNRELATED TO THE ORIGINAL COMPLAINT, THE BATTERY COMPARTMENT WAS NOTED TO BE CRACKED.	No code assigned
5980174	56056538	ON (B)(6) 2016, THE REPORTER CONTACTED ANIMAS, ALLEGING A CALL SERVICE ALARM (CALL SERVICE ALARM ISSUE) ISSUE. THERE WAS NO INDICATION THAT THE PRODUCT CAUSED OR CONTRIBUTED TO AN ADVERSE EVENT.	2506 - Structural problem

MDR_REPORT_KEY	MDR_TEXT_KEY	FOI_TEXT	PROBLEM CODE IN MAUDE
		THE PUMP WAS RETURNED FOR INVESTIGATION AND **A LANGUAGE FILE CORRUPTION** AND A CRACKED BATTERY COMPARTMENT WERE FOUND.	

Table 36: Sample of Computing Technology-related Events with Missing or Incorrect Codes in MAUDE

4.9.3 Implications

Based on our sample data, it is clear that the current problem code assignment in the MAUDE database as published by the FDA is problematic for problems related to computing technology. The current scheme appears to either miss classification, or inaccurately/insufficiently classify events related to computing technology at an alarmingly significant rate (89%). Additional research is needed if this apparent issue also extends to other problems not related to computing technology.

One potential implication of this finding is that any existing research that is based on results from the FDA's MAUDE Search Engine (which is the case with almost all existing research using MAUDE) is potentially invalid if the research uses Product Problem field of the Search Engine highlighted on the screenshot in Figure 24:

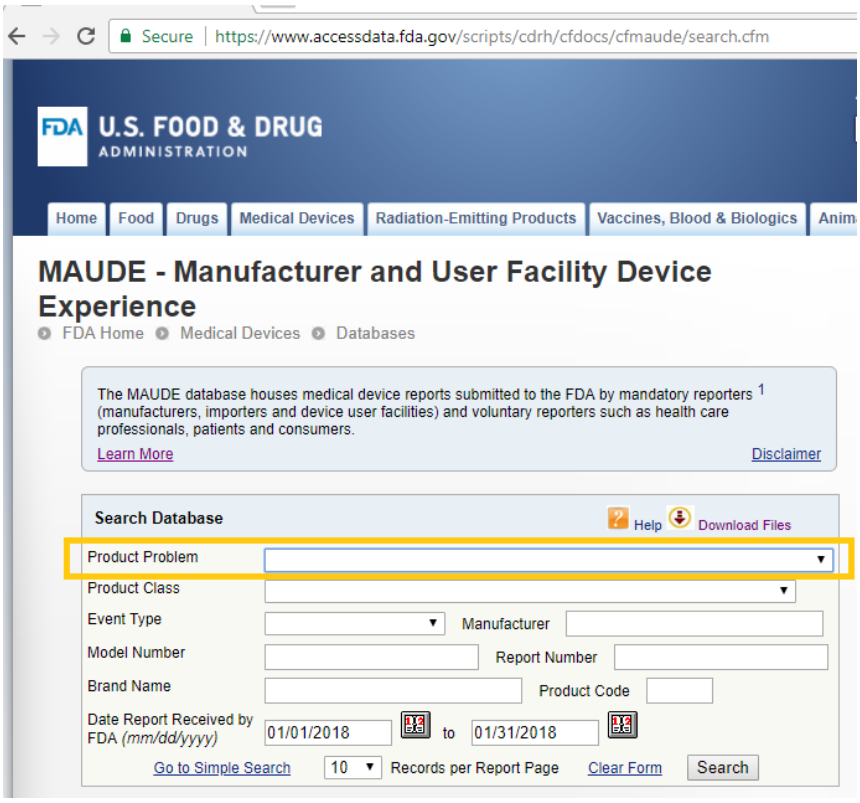


Figure 24: FDA's MAUDE Search Engine

The Product Problem field on the MAUDE Search Engine is a closed, single-selection dropdown list of predefined problem codes. Selecting a value in this field will only match event records that are assigned the selected problem code. Figure 25 shows a section of this dropdown list:

Figure 25: Problem Code Field in MAUDE Search Engine

To illustrate the problem, we used the MAUDE Search Engine to locate some of the records in our sample in Table 36. We found out that using the Product Problem field, there is no way to find the records that do not have a problem code assigned yet. For example, we tried to look up the following record, which per the narrative, is an instance of software failure:

MDR_REPORT_KEY: 2359547
REPORT NUMBER: 6000001-2011-39844
REPORT DATE: 12/06/2011
NARRATIVE TEXT:
AFTER FURTHER INVESTIGATION BY THE QUALITY ENGINEERS, THE **ROOT CAUSE OF THE REPORTED CONDITION WAS ASSIGNED TO SOFTWARE FAILURE**. SOFTWARE ISSUES ARE NOT ASSOCIATED WITH HARDWARE MALFUNCTIONS. THEREFORE THERE WERE NO REPAIRS MADE TO CORRECT THE REPORTED PROBLEM. A SERVICE HISTORY REVIEW REVEALED THAT THIS DEVICE WAS PREVIOUSLY SERVICED FOR FAILURES/PROBLEMS THAT WERE SAME AS OR SIMILAR TO THE CURRENT PROBLEM.

We entered the value of 'Computer software issue' in the Product Problem field, and the value of '6000001-2011-39844' in the Report Number field, as shown in Figure 26:

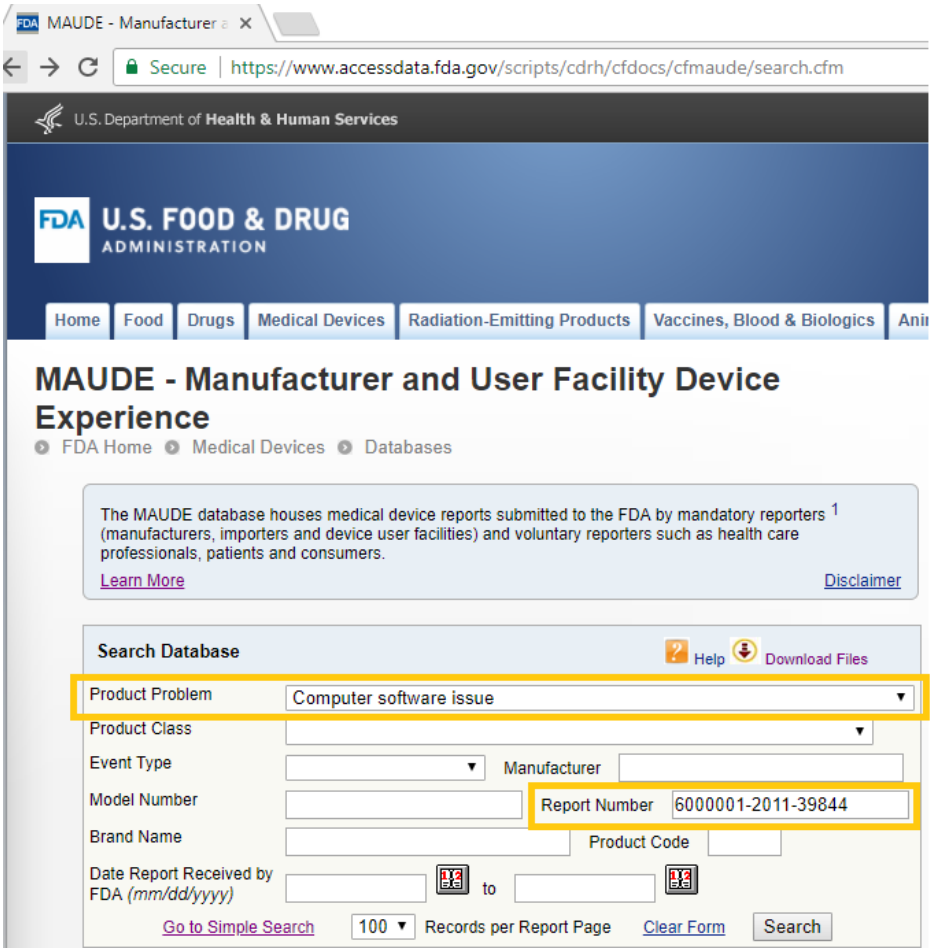


Figure 26: FDA's MAUDE Search Engine Search by Product Problem

The search yielded no results, as shown in Figure 27:

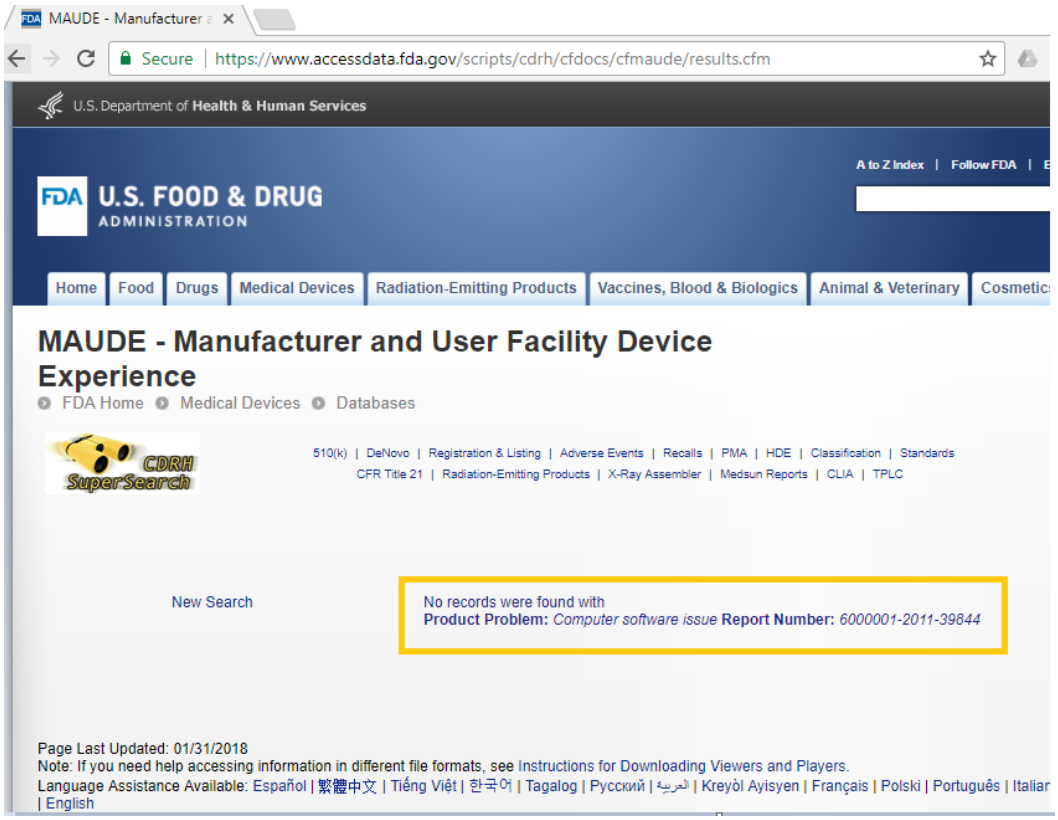


Figure 27: FDA's MAUDE Search Engine Results by Product Problem

We then changed the search parameters to remove 'Computer software issue' from the Product Problem field, and only searched by the Report Number as shown in Figure 28:

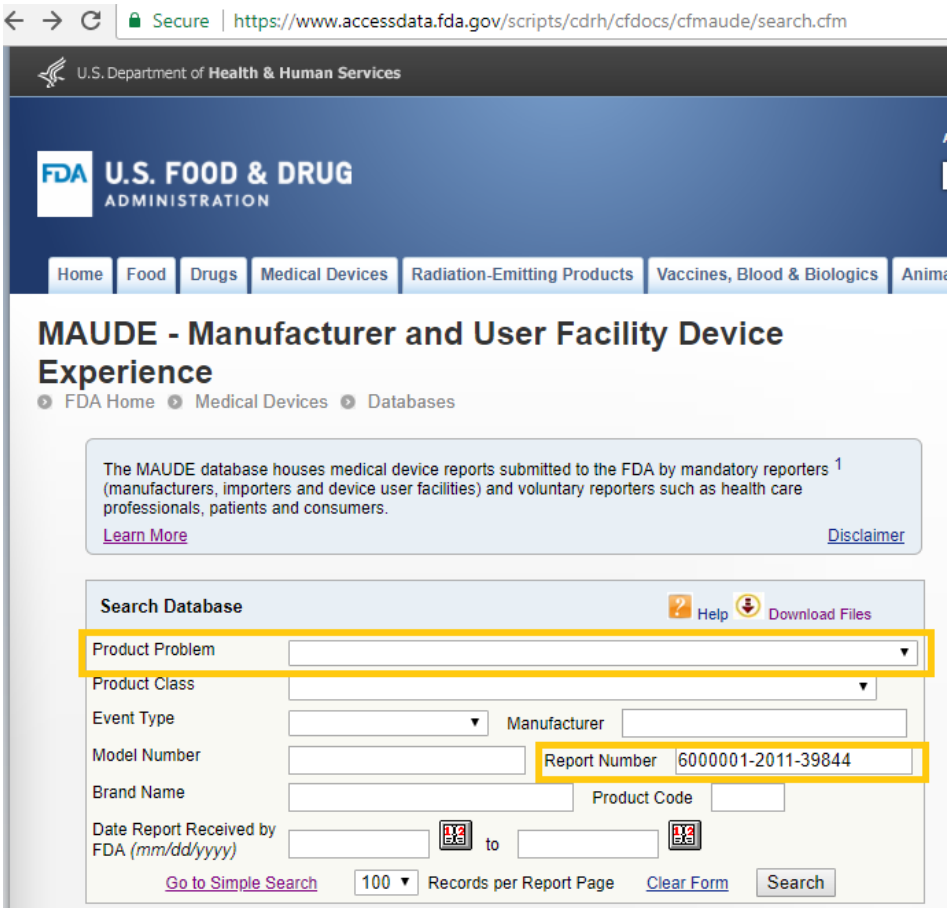


Figure 28: FDA's MAUDE Search Engine - Search by Report Number

This search (without the Product Problem specified) found the record as shown in Figure 29.

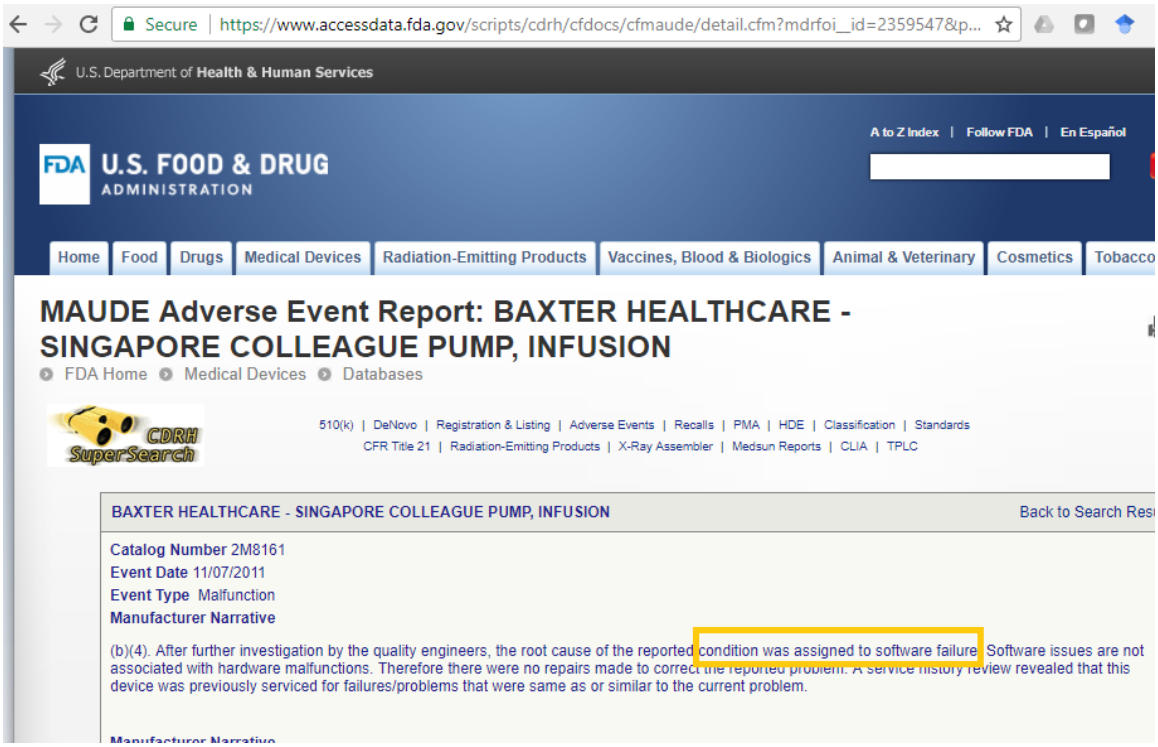


Figure 29: FDA's MAUDE Search Engine - Results by Report Number

These observations confirmed one of our claims that the MAUDE database is missing problem code assignment for many events related to computing technology. Upon further experimentation with the Search Engine, we were also able confirm the cases of incorrect or insufficient problem codes, as shown in Figure 30 - Figure 33:

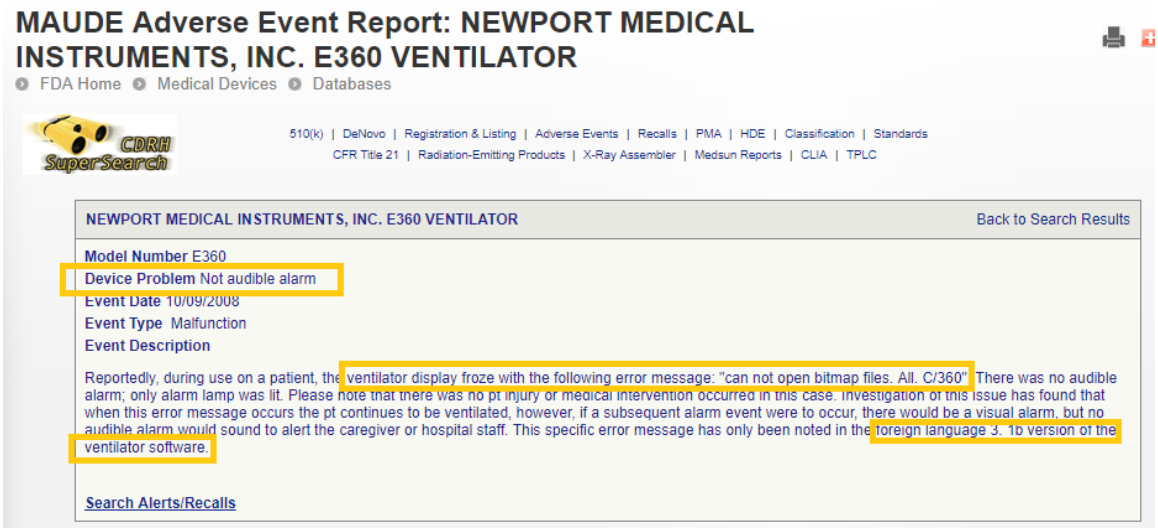


Figure 30: FDA's MAUDE Search Engine – Misclassification Example 1

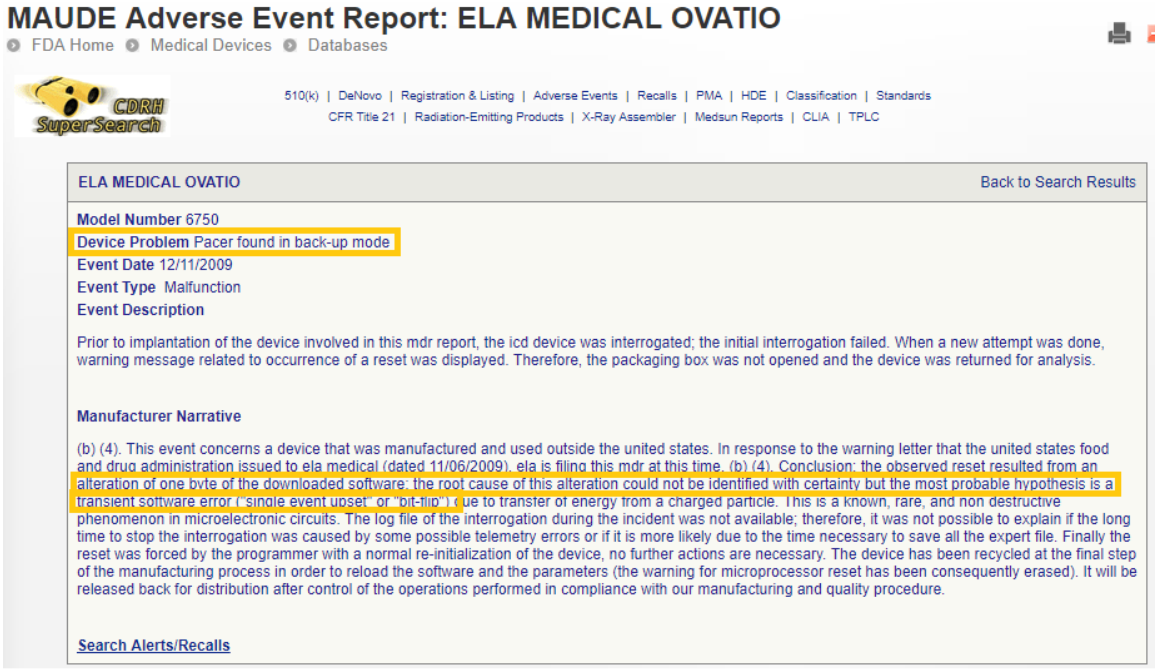


Figure 31: FDA's MAUDE Search Engine – Misclassification Example 2

Figure 32: FDA's MAUDE Search Engine – Misclassification Example 3

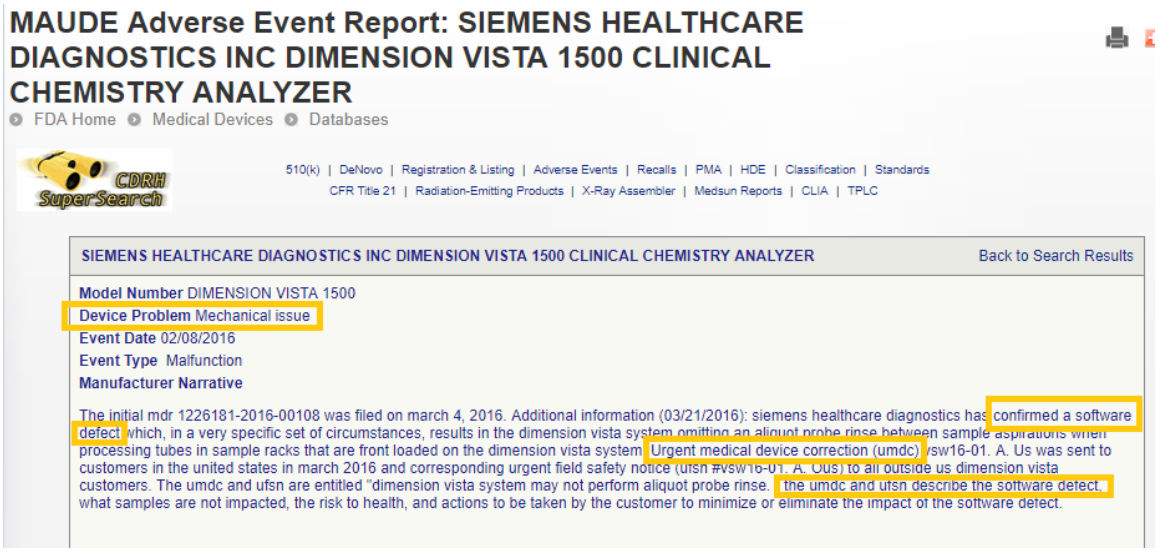


Figure 33: FDA's MAUDE Search Engine -- Misclassification Example 4

Our analysis shows that the search by Product Problem field on the MAUDE Search Engine does not return all events (due to missing problem code assignment) a researcher may have intended to retrieve. Also, in many cases, the Search Engine may return wrong results (due to incorrect or insufficient problem code assignment). The issue, however, is not

with the Search Engine itself (the raw datasets also have the same problem as we already discussed in the previous section), but rather in the underlying scheme by which the problem codes are assigned to events and published through MAUDE.

4.9.4 A Sample of Published Studies Potentially Impacted

Because problem codes in MAUDE are assigned by submitters of medical device event reports, the accuracy of these codes is highly dependent on the ability of the submitter to navigate a large and complicated dictionary of codes. The FDA does not currently appear to curate this information before it is published to the public. In the case of computing technology-related events, we have demonstrated that the assigned problem codes in MAUDE are highly inaccurate and cannot be trusted. Considering the root causes (i.e. submitter decides the codes, FDA simply publishes the reports back without any verification), there is no reason to believe that this issue is limited to only computing technology related causes.

In particular, we suspect that all existing research that depended on the accuracy of the Product Problem field in the MAUDE Search Engine is impacted and potentially invalid. We performed a cursory search of the Internet on published studies and found several that used this field in their research method. Below, we provide a small sample of these studies:

(Galhotra, Amesur, Zajko, & Simmons, 2007) queried the MAUDE Search Engine for Product Problem of "migration", Brand Name of "Günther Tulip", manufacturer of "Cook" and Date Report Received by FDA of "01/01/2000 to 05/29/2007", looking for reported incidents of Günther Tulip filter migration. The study was published on the *Journal of Vascular and Interventional Radiology* Volume 18, Issue 12.

(Sfyroeras, Koutsiaris, Karathanos, Giannakopoulos, & Giannoukas, 2010) used the MAUDE Search Engine to find incidents of carotid stent fractures. They used the Product Problem field ("fracture" and "break") to identify incidents of their interest. A portion of the resulting records were used in the subsequent analysis. This study was published on the *Journal of Vascular Surgery*, Volume 51, Issue 5.

(Kramer et al., 2012) searched the MAUDE database using the FDA's Search Engine looking for adverse events related to security and privacy problems. They evaluated the approximately 1,000 problems listed in the Product Problem field for a plausible relationship with security or privacy and performed the search using those problem codes. The study was published on *PLOS One* Volume 7, Issue 7.

(Alemzadeh et al., 2013) used the MAUDE Search Engine to query medical device adverse sevents. They used the Product Problem field to identify computer related adverse events. The research also relied on the Product Problem as the *cause* for the reported adverse events. The study was published on *IEEE Security & Privacy* Volume 11, Issue 4.

(Kwazneski, Six, & Stahlfeld, 2013) used the MAUDE Search Engine looking for incidents of laparoscopic stapler malfunction. One of the fields they used in their query was Product Problem, which they set to 'misfire'. The study suggested potential underreporting of the incidents based on the results the Search Engine returned, but it nevertheless relied on the Product Problem field to be accurate in the classification of the reports. This study was published on *Surgical Endoscopy* Volume 27, Issue 1.

The list presented above is a small sample of a potentially larger number of published studies that may have relied on the Product Problem field of the FDA's MAUDE Search Engine. It is not possible to quantify exactly how these studies may have affected decision

making by clinicians or the public. However, based on the evidence presented in this and the preceding sections, it is clear that the Product Problem field in the FDA's MAUDE Search Engine presents a significant risk of misinformation.

CHAPTER V: DISCUSSION

5.1 RELATIONAL RESEARCH

In this research, we have demonstrated that a significant number of medical device failures are computing technology related. It is important to emphasize that our research was relational and not causal. In other words, we did not seek to find medical device failures *caused* by computing technology. Instead, we focused on identifying failures with some form of interaction or relationship with the onboard or supporting computing technology. This relationship may have been causal or associative.

The primary reason why we pursued a relational approach to our investigation was because we recognized very early in our research that it would not be possible to reliably determine causal relationship between a medical device failure and computing technology. There are three main reasons why such determination is difficult:

1. **Inconclusive data:** The data in the Manufacturer and User Facility Device Experience (MAUDE) database contains reports of events submitted by various manufacturers, distributors, healthcare facilities and the general public. As such, the data in this database is highly diverse in terms of its fidelity and often lacking conclusive evidence necessary to establish definitive causal claims.

2. **Computing technology as vehicle**: The role of computing technology in the manifestation of a medical device failure can be as a reporting function. For example, even a confirmed observation of a 'software error message', does not necessarily mean a software defect. Software is sometimes the means by which various failure states (may or may not be computing technology-related) are conveyed to the User.
3. **Blurring Hardware/Software Divide:** Medical devices can be very complex systems with multiple computers onboard. Components in a device system can be fitted with digital sensors, integrated circuit boards with powerful processors, and firmware. Therefore, it can be near impossible to identify a root cause simply based on a complaint of a functional deficiency. For example, a report of a "hardware malfunction" does not necessarily rule out the role of computing technology. If the hardware contains embedded firmware, there is potential that the failure may be due to a defect in the firmware.

5.2 OBSERVATIONS

The nearly 7-fold increase in the number of computing technology-related medical device events reported to the U.S. Food and Drug Administration (FDA) between 2007 and 2016 appears to be roughly in line with the similar growth (5-fold) in the overall number of events reported in the same period. However, computing technology-related events since 2012 were reported at a much higher rate compared to the overall events. Between 2012 and 2016, the computing technology related events reported to the FDA grew by 258%, whereas the overall events grew by 78%. It is not clear from the available data what explains this

difference in the growth, but it is conceivable that more medical devices equipped with computing technology entered the market in the more recent years.

Another observation we made was that nearly all (99.36%) of the reports of medical device failures related to computing technology were submitted by manufacturers of the devices. Contrary to our expectations, patients submitted only 0.18% of the computing technology-related events to the FDA. This, however, does not mean that patients were disengaged. In fact, we observed that patients were the original reporters (even though not the submitters) of a significant number (32.46%) of the events ultimately submitted to the FDA. This suggests that patients tend to report issues and adverse events to manufacturers of the devices instead of the FDA. It is not obvious from the available data why patients and the general public do not submit reports of failure at a higher rate. One possible explanation is that unlike medical device manufacturers, patients are not *required* by law to submit the reports and may not even be aware of the FDA's Medical Device Reporting (MDR) programs.

We have shown in this research that computing technology-related medical device events can have serious implications on patients, including death. We have also shown that the rate of these adverse events has remained fairly constant over the study period. It is not entirely clear from the MAUDE data why the rate of adverse patient events has not followed a trend in either direction over the 10 years. One possible explanation for this relative stability may be that the FDA's premarket and post-market regulatory requirements are effective in preventing entry of unsafe devices to market and that the current rates (consistently less than 1% death events; less than 15% injury events) represent a state of equilibrium between safety (outcome) and constraints (regulations).

This research has also demonstrated that the current scheme of problem code assignment in the MDR submissions significantly masks computing technology-related problems. Although outside the scope of our research, there is no reason to believe that this issue is limited to computing technology-related problems only. The problem code assignment, in general, appears to be dependent on the ability of the submitter of the event report to navigate the FDA's problem code tables and inconsistent at best. It is also not clear how much, if at all, the FDA curates the code assignment before the information is published to the public. Because downstream research based on flawed data can have serious implications, we recommend that the FDA consider removing the Product Problem field from its Search Engine until the issues we highlighted have been resolved. We also recommend that the FDA consider implementing a machine learning-based approach to assigning problem codes using the narrative records. In our experience, such a method may be significantly more effective and accurate than the status quo.

5.3 LIMITATIONS

This research has several limitations. First, the data for this study, although fairly large in size (more than 5 million records of medical device failure events and more than 11 million narrative records related to those events), comes from one source: the MAUDE database. The representativeness of this data depends on how well it reflects the prevalence and types of medical devices failures. Due to the unavailability of another dataset with similar characteristics, the study could not be cross-validated with an independent reference dataset.

To identify computing technology-related failure events, we used the narrative text records submitted with the reports of medical device failure. As with any automated text classification task, our classification is subject to grammatical idiosyncrasies and semantic

drifts in the natural language. We also assumed the information in the narratives to be correct, as there was no way to conclusively confirm the validity. Even in cases of conflicting narratives on a given event, we assumed any complaint of a malfunction to be correct. For example, in several instances, we found narratives such as "the reported firmware error was not observed…. the device performs per the specification." We classified each of such instances as positive because we assumed that the reporter of the event witnessed the error, even if the investigator was unable to reproduce it on a follow up.

The training data for our machine learning experiments was generated using a snowball-type relation extraction scheme. The seed data for this scheme was extracted from random locations in the corpus and entirely verified by a human expert. However, the seed data only constituted a very small portion (2,449 records) of the corpus (11,689,482 records). In order to make our training data more comprehensive, we adopted a closely supervised but semi-automatic snowballing scheme, which yielded a more representative set (145,701 records) from the seed data. While we made every practical effort to maximize the accuracy of the labels in the training data (such as through manual as well as automated quality control processes discussed in 3.2.4.2.2) and minimize the effects of any residual, potentially misclassified records (such as through stringent classification criterion discussed in Section 3.2.6 and multiple methods of verification as discussed in Section 3.4), not all training data generated through the seed-based snowballing method was manually verified in its entirety by a human expert.

While we have attempted to provide annual breakdown of numbers where possible, the overall aggregate numbers reported in the study assume the reporting environment as constant. We have not accounted or adjusted for any political or regulatory events that may

have caused reports of failures to be higher or lower in some years. Our approach to examine MAUDE data over 10 years was partly in consideration for potential transient swings (such as in the case of year 2007, where one manufacture submitted a disproportionately large number of failure reports), but it should be noted that inferences made in our research assume the data in MAUDE to be, on average, representative of the state of medical device failures over the study years.

We also did not account for any updates made to the narrative records or the event reports submitted to the FDA after the original submission. We assumed the original submission to be correct. We also assumed that the events reported to the FDA were not normally expected use-related errors. Since almost all of the events were submitted by manufacturers, we assumed that the reported events constituted an abnormal device behavior that the submitter determined to be reportable.

Finally, we also did not filter out or account for multiple reporting (if any) of the same issue that may have manifested at multiple sites. We counted each unique report of a medical device event published by the FDA in the MAUDE database as one medical device event for our study. There was also no reliable way in the published MAUDE dataset to identify and eliminate any duplicates and a manual approach would be impractical for over 5 million event records. Because our focus was to study the *reporting* of failures, and not to identify the implicated devices, where eliminating potential duplicate submissions may have been more important, we assumed a uniform distribution of any duplicate submissions to the FDA, and not material to the results in our study.

CHAPTER VI: SUMMARY AND CONCLUSIONS

6.1 SUMMARY

The adoption of computing technology in modern medical devices is ubiquitous and growing. A typical hospital bed is surrounded by computerized instruments monitoring the patient and delivering therapy. Software technology has transformed medical imaging in pathology, radiology and surgery. Even implantable devices such as defibrillators and pacemakers are powered by embedded computers. Clinical charts are increasingly in the electronic form and integrated with other computer systems. Computer systems in modern analyzers automate test processing in clinical laboratories. In recent years, the proliferation of wearable devices and software applications that run on smartphones has expanded the traditional boundary of medical devices from clinical instruments to consumer gadgets.

Intrigued by the role computing technology plays on medical devices, we sought to understand if/how computerization had impacted safety and efficacy of medical devices. More specifically, we wanted to explore what portion of medical device failures were potentially computing technology-related and whether those failures could cause serious harm on patients.

We performed an extensive review of existing literature on the topic and determined that the existing body of research was lacking on the safety of medical devices. Existing research was particularly scarce on computing technology in medical devices.

The U.S. Food and Drug Administration (FDA) collects reports of medical device failures as a part of its Medical Device Reporting (MDR) program. Manufacturers of medical devices are required by law to report events of device failures to the FDA. The reports of medical device failure events collected through the MDR are published by the FDA for public use through a database known as Manufacturer and User Facility Device Experience (MAUDE).

The MAUDE database contains millions of reports of medical device events since the early 1990s. Each report in the database contains information on the event, one or more problem codes associated with the events, device information, patient outcome information and descriptive narratives.

For our research, we decided to examine the MAUDE database for medical device events reported between 2007 and 2016 (10 years) around the topics of our interest. So, we downloaded a set of very large raw data files from the FDA's MAUDE Website and imported them into a sophisticated database system for analysis.

To identify medical device failure events in MAUDE related to computing technology, we first used the problem code information associated with each event in the database. The problem codes are codified description of functional deficiency and part of the FDA's Device Problem Code Hierarchy (DPCH). It was unclear how much the FDA curated the problem code assignment on a submitted report before the report is published on MAUDE,

but we found that it was a widely utilized field. We also found that many of the problem codes were related to computing technology.

Analysis based on the problem codes, however, showed that only negligible (0.4%) portion of the medical device failure events reported to the FDA were computing technology-related. This did not align well with our assumptions that computing technology played a central role in modern medical devices. Upon further investigation of the MAUDE database, we found that the problem code field that we used in our analysis may not be a reliable attribute to identify computing technology-related events.

Based on the observations of the preliminary research, we decided to pursue an alternative approach to identify computing technology-related events. For this, we found that the descriptive narratives that were also part of the MAUDE database contained more reliable information that could identify an event as potentially computing technology related.

There were 11.69 million records of natural-language narratives for 5.11 million reports of failure events in the MAUDE database for the 10-year study period (2007 - 2016). Manually reviewing each narrative to determine whether it is related to computing technology would not be practical. We needed an automated way to perform this task, and solving this problem constituted a major portion of our research.

To classify the narrative records, we explored machine learning techniques. We determined that a supervised machine learning approach would be a suitable choice for our [classification] problem. However, there was no preexisting training data available to train the computer models.

To generate the training data necessary for the machine learning-based classification, we started with a small set (N=2,449) of seed data randomly extracted from the existing

narrative records and verified by a human expert using an interactive game-like program that included a decision support feature. However, we felt that this sample size would be too small to train the models to classify a corpus of 11.69 million records. We recognized that we needed make the training data more *generative* than this small set.

To generate a more comprehensive set of training data, we followed a snowball-type relation extraction scheme, in which a machine learning classifier would initially learn from the seed data and classify new narrative records. Records meeting a set of predefined probabilistic thresholds (p=0.9) would be qualified to be part of the training data set, pending quality control (QC) by a human expert. The classifier, would then train on the expanded training data set to make new classifications. We wrote a program to implement this multi-batch, QC-integrated iterative relation extraction process. Given the size of the corpus, our focus on this exercise was on producing more *generative* training data.

After a series of automated sessions to generate training data supplemented by human corrections, we produced a larger set (N=145,701) of records that we found in our sampling to be sufficiently generative. These records were randomly extracted from all files (10 total, each containing records for one year) in the corpus, with at least 1% of records from each file. This training data comprised of roughly half *positive* (i.e. computing technology-related) and half negative records.

Using the training data generated, we trained classifiers based on three machine learning models: Naïve Bayes, logistic regression and support vector machine (SVM) with stochastic gradient descent learning. We then developed a program to classify each of the 11.69 million narrative records using a unanimous voting scheme where each of the three

classifiers needed to classify a record as positive with a very high probabilistic confidence (p=0.95, except for SVM which does not offer a probability score).

This machine learning-based approach classified 1.53 million of the 11.69 narrative records in MAUDE as computing technology related. We verified the performance of the classification with standard statistical measures, automated checks as well manual sampling. Our overall classification had the precision of 0.97, recall of 0.87 and the F_1 score of 0.92. In a manual verification of 1,000 randomly sampled positive classified records, we found that 962 (96.2%) records were correctly classified.

We then joined the results of this classification with other data available in MAUDE to perform a series of analyses around our research questions. These included, the overall and yearly trend analysis, analysis by submitters and reporters of events, and events associated with adverse patient events such as injury or death.

We also tested, through sampling, and the FDA's Search Engine if the problem code assignment in MAUDE was indeed masking computing technology-related problems and found some evidence that could potentially invalidate some of the existing peer-reviewed studies using the MAUDE Search Engine.

6.2 CONCLUSIONS

Between the years 2007 and 2016, there were a total of 5,110,200 reports of medical device failure events reported to the FDA through the Medical Device Reporting (MDR) provision of the United States Code of Federal Regulations (21CFR803.19).

Using the machine learning-based classification approach, we identified 1,155,516 (22.61%) of the 5,110,200 reports of medical device failure to be *related* to computing

technology. We also found that computing technology-related medical device events were on the rise year-over-year on an absolute basis with a near 7-fold increase between 2007 and 2016. This *confirms our hypotheses that computing technology is related to a significant portion of medical device failures reported to the FDA* and also that *computing technology is related to increasing number of medical device failures*. This observation is also roughly in line with the FDA's analysis using a separate database which found that 15% of all medical device recalls between 2008 and 2012 were due to software-related causes (FDA, 2014b).

Nearly all (99.36%) of the reports received by the FDA between 2007 and 2016 were submitted by medical device manufacturers. While manufacturers were the *submitters* of the reports, 32.46% of the events were discovered and reported [to manufactures] by the general patients. The general public (including patients) was also found to be the original reporters of a higher percentage of computing technology-related events (33.13%) compared to the overall events (13.52%) in the 10-year period.

We also found that a total of 3,449 patient deaths were associated with medical device failure events related to computing technology in the 10-year period (2007 - 2016) *confirming our hypothesis that medical device failures related to computing technology can have fatal consequences on patients*. While the number of deaths were on the rise in almost each of the 10 years on an absolute basis, it remained consistently low (less than 1%) as a percentage of all computing technology-related medical device events.

A total of 104,754 events of patient injuries were also found to be associated with computing technology-related events in the same period (2007 – 2016) with a steep growth year over year in the latter years (four-fold increase between 2013 and 2016). Overall, patient

injuries were associated with 9.07% of all computing technology-related events in the 10-year period.

We also found that the current scheme of assigning problem codes to MDR events was significantly masking computing technology-related problems. Based on a sample of 102 medical device failure events in MAUDE with a computing technology-related cause, we found that a vast majority (89%) either did not have any problem code assigned (63%) or were assigned a code not related to a computing technology (26%). This *confirmed our hypothesis that the problem codes assigned to medical device failures in the MAUDE database are masking problems related to computing technology*. We also confirmed our finding of the lack of problem code assignment and misclassification using the FDA's MAUDE Search Engine and demonstrated how the FDA's MAUDE Search Engine is providing incomplete or inaccurate results that may be impacting downstream research.

6.3 RECOMMENDATIONS FOR ACTION AND FURTHER RESEARCH

As we stated at the outset, the overall purpose of this research is to help improve the quality of healthcare through safer and more reliable medical devices. We hope that the findings of this study have contributed new insights to our understanding of the state of computing technology in medical devices and that these insights will help make medical devices safer and more reliable. Toward this goal, we offer some specific recommendations to the FDA, the medical device industry and the research community based on our experience and findings in this research.

6.3.1 Recommendations to FDA

Conceptually, the FDA's MDR program has a tremendous potential in improving safety and reliability of medical devices. The idea of collecting reports of medical device

failures and using this information in improving patient safety has merits. However, it is unclear exactly what practical purpose the current MDR systems and processes are serving and whether they are contributing to the intended goals.

First of all, the current system of collecting data on medical device events appears to be onerous and evident by the inconsistencies reflected in the MAUDE database. The FDA currently collects reports medical device events through a set of MedWatch forms (see Appendix A and B) that are complicated and require mastery of additional dictionaries and coding manuals on the part of submitters, particularly, for mandatory submissions. We recommend that the FDA simplify this reporting system. For example, a new reporting system could be entirely online, not simply as a Web version of the existing forms, which already exist, but rather as a dynamic application that collects only relevant data based on the types of medical devices and failures instead of the current one-size-fits-all approach. The tens of millions of existing records in the MAUDE database could be used in the development of the new system to design a taxonomy of various devices and relevant data attributes.

We also recommend the FDA to ensure the quality of data in the MDR submissions. The current approach appears to suffer from the colloquial *garbage-in-garbage-out* problem. In other words, the quality of data received through the MDR is very dependent on the ability of the submitter to navigate a complex hierarchy of problem codes and clinical information. We have highlighted this problem in our research, where we found a significant number of records either not containing problem codes or containing insufficient or incorrect codes. FDA should evaluate if it is possible to identify and correct inaccuracies in the existing data and implement measures to ensure higher data quality on new submissions.

While the FDA disclaims accuracies in the MDR data, it nevertheless, currently feeds it back to the public through the MAUDE database, which a large number of existing studies have used as the source of truth. It is conceivable that the researchers behind those studies are not fully aware of the flaws in the data, since most of the studies appear to have been conducted using the FDA's MAUDE Search Engine, which masks some of the problems. In particular, we recommend the FDA to immediately remove the "Product Problem" as one of the primary fields on its MAUDE Search Engine, as we have demonstrated through this research that this field has the potential of yielding highly inaccurate results. Although outside of the scope of our study, some of the other fields may also have similar issues, and the FDA should evaluate the risks associated with their use on the Search Engine.

Another recommendation we would like to offer to the FDA is to consider adopting alternative methods of classifying MDR events, particularly for problem code assignment. The FDA's current direction appears to be to build a "harmonized" taxonomy of various problem codes and require submitters to follow this taxonomy to find and assign appropriate codes. Unifying the various dictionaries that exist around problem codes into one harmonized taxonomy is a commendable effort. However, this approach suffers from two problems. First, it is near impossible, however large the dictionary is, to comprehend every possible problem into a *closed* dictionary without losing precision. This is particularly true for domains that are newer or evolve at a very high pace, such as computing technology. For example, the current set of codes associated with computing technology in the FDA's problem code dictionaries is severely lacking in both breadth and depth.

The second problem is that the harmonized problem codes do not necessarily take away the complexities for the submitter, who still has to be able to navigate the new

dictionary. Therefore, we do not believe that it will have a significant effect on the overall of quality of data received in the MDR submissions. One suggestion is for the *FDA,* instead of the submitter, to perform the assignment of codes based on all other information in the submission. Another suggestion is to leverage automatic classification techniques, such as machine learning to automate or aid the human expert performing the problem code assignment.

We also recommend the FDA to engage and share with public what it does with the data it receives through the MDR. Currently, the whole system, from an outside researcher's perspective, appears to be a massive, passive database of minimally curated raw information. We recommend the FDA to analyze and present this data to the public in a more synthesized representation, such as annual reports. This type of engagement may also promote voluntary participation by the public in the MDR process, which based on our research findings, is currently very low.

Our recommendations to the FDA in this section specifically centered around the MDR program. This was because we believe that there are significant opportunities for improvement in this program based on our experience with the MDR data in this research. We also performed a survey of the FDA's current pre-market and post-market guidance to the industry around medical device software and did not find any obvious gaps. However, we should note that we found the FDA's current position on medical device classification of clinical software systems (e.g. is electronic health records system a medical device?) unclear and ambiguous at best. We recommend the FDA to clearly establish its regulatory framework for *software as a medical device* type of systems in a way that fosters innovation without compromising patient safety.

6.3.2 Recommendations to the Industry

We have shown in this research that computing technology is related to a significant number of medical device failures and that such failures can have serious adverse patient events. At a high level, these findings are not likely to be a surprise to the industry with a deep empirical domain knowledge of how modern medical devices are built and the first-hand knowledge of their failures. However, we hope that this research provides the industry with a formal baseline on computing technology-related failures and a rationale for solidifying investments in ensuring higher quality computing technology artifacts that are integrated into medical devices.

In addition to the objective analyses we performed and discussed in the previous sections, we also examined thousands of narrative records in the MAUDE database in the course of this research. While not easily representable in numeric forms, there were a number of insightful observations we made in the course of our review of these records. Below, we provide some recommendations to the industry based on the findings of our research and the cases we reviewed.

We have found in this research that computing technology-related medical device events reported to the FDA are on the rise on an absolute basis. There is not enough data to show objectively why this is the case, but existing literature suggests that medical devices are getting increasingly more sophisticated and complex in their software capabilities. We recommend the medical device industry to be aware of this trend and incorporate sound software engineering practices into the medical device development process.

Controlling risks associated with computing technology-related modes of medical device failures can be challenging. This is because computing technology, such as software

can be abstract and its behavior not always deterministic to a functional user. Often times, hidden bugs escape verification and manifest in production in unpredictable and sometimes intermittent fashion. In the course this research, we found countless instances of reports where the manufacturer was unable to reproduce an abnormal device behavior clearly observed by the User. These cases suggest that there is an opportunity for manufactures to implement a more thorough trace of events and User actions in the medical device software. Analyses of such traces could iteratively feed back into the assessment of risks, mitigation as well as design improvements.

Another common issue we observed was on the retention of the diagnostic data on medical devices. In many cases, root cause investigations by manufactures were inconclusive due to the deletion of relevant diagnostic data from the device to make room for new data. While this may be a challenge for small form factor systems with very limited storage such as implantable and wearable devices, larger devices should allocate persistent storage for important diagnostic data with a longer lifecycle.

With the advancements made in computing and network technologies in the recent years, the medical device industry should also consider transitioning to a more proactive method of understanding and preventing medical device failures. As evident in this research, the current method of discovering a medical device failure is passive. In the cases we studied, the process of a defect discovery overwhelmingly starts with a user complaint, followed by a root cause investigation by the manufacturer. This makes us wonder how many failures go unreported or not reported in time to prevent adverse patient events. Technologies are available today to provide near real-time monitoring of computer and software systems remotely. Considering how connected modern medical devices are, such technologies could

be integrated into medical devices to provide a more proactive framework for detecting and preventing medical device failures.

Usability and human-device interaction is another area where believe the industry could improve. In many cases we studied, user complaints were dismissed in subsequent investigations as expected system behavior by design. This suggests a disconnect – what the manufacturer may have considered by design is perceived by the User as an abnormal system behavior. We believe that a greater emphasis by manufactures on human factors and usability could help reduce medical device failures.

It is clear from the volume of data in the MAUDE database that the medical device industry is highly engaged in the MDR process. This may be because manufacturers and distributors of devices are *required* by law to submit MDR reports to the FDA, but the strong participation by the industry has, nevertheless, resulted in a large amount of information to be collected, which could be used to generate insights as we tried to do in this research.

However, the medical device industry should emphasize the quality of their MDR submissions. In many cases, we found the submissions to be merely for compliance purposes lacking proper care for the integrity or accuracy of data contained in the reports. The fidelity of submitted data was particularly questionable for medical device events related to computing technology, as we have highlighted in this research. In other cases, we found the submissions to be intended to contest or disclaim a failure. Such a compliance or liability-focused approach can mislead downstream aggregation and reports and may not contribute to the overall, shared goal of making devices safer. We recommend the industry to emphasize objectivity and quality of data in their MDR submissions.

6.3.3 Recommendations for Further Research

In this research, we provided a baseline for our understanding of the state of medical device failures related to computing technology. There are a number of opportunities for follow up research on this topic that we could not get to for time and other reasons. Below we present some suggestions for the research community.

The design of the machine learning-based text classification experiment in our research used dichotomous classes as outcome variables. A follow-up research could extend this with a multinomial classification approach where the classes could be the different types of medical device failures. Objective information on the most prevalent types of failures could help the industry focus on addressing those and make medical devices safer and more reliable. There is also an opportunity for an unsupervised clustering experiment that can identify the most significant patterns in the MAUDE narrative data.

There is also an opportunity to design targeted machine learning experiments using the MAUDE narratives on specific topics. One topic, which we were interested in is cybersecurity. There is an abundance of anecdotal evidence and industry focus on the need to enhance cybersecurity in medical devices. Considering the risks, we feel that such emphasis is rightly placed. However, there is a lack of objective evidence on the state of cybersecurity risks and controls in medical devices. The structured data captured through the MDR does not have enough resolution to provide metrics on this topic. The unstructured narrative data, on the other hand, could provide important insights on our understanding of cybersecurity issues in medical devices.

Another area of follow on research could be to find answers to some of the open questions in our research. For example, we have shown that medical device failures related

to computing technology are on the rise on an absolute basis, but it is not clear from the MAUDE data whether such trend is due to less reliable computing technology, or simply greater number of devices in the market with computing technology. Similarly, we have shown that adverse patient events associated with medical device failure related to computing technology are increasing, but it is not clear whether that is simply a reflection of more computer-equipped devices in use. There is an opportunity to help answer these questions by correlating our findings with another data source or research that may have information on other relevant factors, such as prevalence.

We have demonstrated in this research that the current scheme of problem code assignment in the MAUDE database is masking computing technology-related events at a significant rate. We believe that this issue extends beyond computing technology-related problems, but a recommendation for future research is to objectively investigate whether that is the case. Such a discovery could help make a stronger case to the FDA for an overhaul of the current scheme.

There is also an opportunity for innovators to create online applications with key performance indicators (KPIs) and visuals based on the MAUDE data. For example, it would highly useful for the public and the industry to understand the trends in the MAUDE data on a real-time basis. The types of KPIs and the analytics that could be performed on this dataset are wide-ranging. These, for instance, could include problems by device, adverse events by device, reports by sources, problem trends by device, etc. These KPIs could help predict potential problems and minimize adverse patient events.

Finally, we also recommend further research on how the data in MAUDE on medical device failures correlates with data from FDA's other post-market surveillance

programs. For example, a topic for investigation could be to find out how many and what types of the events in MAUDE end up resulting in a product recall, which are tracked separately by the FDA in a different database. A strong correlation could potentially indicate the data in MAUDE as an early predictor of serious medical device failures and help prevent adverse patient events.

REFERENCES

Agichtein, E., & Gravano, L. (2000). *Snowball: Extracting relations from large plain-text collections.* Paper presented at the Proceedings of the fifth ACM conference on Digital libraries.

Alemzadeh, H., Iyer, R. K., Kalbarczyk, Z., & Raman, J. (2013). Analysis of safety-critical computer failures in medical devices. *IEEE Security & Privacy, 11*(4), 14-26.

Andreoli, J. M., Lewandowski, R. J., Vogelzang, R. L., & Ryu, R. K. (2014). Comparison of complication rates associated with permanent and retrievable inferior vena cava filters: a review of the MAUDE database. *Journal of Vascular and Interventional Radiology, 25*(8), 1181-1185.

Bielefeldt, K. (2017). Adverse events of gastric electrical stimulators recorded in the Manufacturer and User Device Experience (MAUDE) Registry. *Autonomic Neuroscience, 202*, 40-44.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010* (pp. 177-186): Springer.

Boulos, M. N. K., Brewer, A. C., Karimkhani, C., Buller, D. B., & Dellavalle, R. P. (2014). Mobile medical and health apps: state of the art, concerns, regulatory control and certification. *Online journal of public health informatics, 5*(3).

Chen, M. M., & Holsinger, F. C. (2016). Morbidity and mortality associated with robotic head and neck surgery: an inquiry of the food and drug administration manufacturer and user facility device experience database. *JAMA Otolaryngology–Head & Neck Surgery, 142*(4), 405-406.

Clarke, E. M., & Wing, J. M. (1996). Formal methods: State of the art and future directions. *ACM Computing Surveys (CSUR), 28*(4), 626-643.

Cohen, A. M., & Hersh, W. R. (2005). A survey of current work in biomedical text mining. *Briefings in bioinformatics, 6*(1), 57-71.

Connor, M. J., Marshall, D. C., Moiseenko, V., Moore, K., Cervino, L., Atwood, T., . . . Recht, A. (2017). Adverse Events Involving Radiation Oncology Medical Devices: Comprehensive Analysis of US Food and Drug Administration Data, 1991 to 2015. *International Journal of Radiation Oncology* Biology* Physics, 97*(1), 18-26.

Cope, J. U., Samuels-Reid, J. H., & Morrison, A. E. (2012). Pediatric use of insulin pump technology: a retrospective study of adverse events in children ages 1–12 years. *Journal of diabetes science and technology, 6*(5), 1053-1059.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning, 20*(3), 273-297.

Denger, C., Feldmann, R. L., Host, M., Lindholm, C., & Shull, F. (2007). *A snapshot of the state of practice in software development for medical devices.* Paper presented at the Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on.

DiBardino, D. J., McElhinney, D. B., Kaza, A. K., & Mayer, J. E. (2009). Analysis of the US Food and Drug Administration Manufacturer and User Facility Device Experience database for adverse events involving Amplatzer septal occluder devices and comparison with the Society of Thoracic Surgery congenital cardiac surgery database. *The Journal of thoracic and cardiovascular surgery, 137*(6), 1334-1341.

Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM, 55*(10), 78-87.

Erickson, B., & Greenes, R. A. (2014). Imaging Systems in Radiology. In *Biomedical Informatics* (pp. 593-611): Springer.

Everett, K. D., Conway, C., Desany, G. J., Baker, B. L., Choi, G., Taylor, C. A., & Edelman, E. R. (2016). Structural mechanics predictions relating to clinical coronary stent fracture in a 5 year period in FDA MAUDE Database. *Annals of biomedical engineering, 44*(2), 391-403.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., & Lin, C.-J. (2008). LIBLINEAR: A library for large linear classification. *Journal of machine learning research, 9*(Aug), 1871-1874.

FDA. (1999). Guidance for Industry, FDA Reviewers and Compliance on Off-The-Shelf Software Use in Medical Devices. In: FDA.

FDA. (2002). General Principles of Software Validation; Final Guidance for Industry and FDA Staff. In: FDA,.

FDA. (2005). Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices. In: FDA,.

FDA. (2014a). *Content of Premarket Submissions for Management of Cybersecurity in Medical Devices*

FDA. (2014b). *Medical Device Recall Repot FY2003 to FY2012*. Retrieved from http://www.fda.gov/downloads/AboutFDA/CentersOffices/OfficeofMedicalProductsandTobacco/CDRH/CDRHTransparency/UCM388442.pdf

FDA. (2015). *What does it mean for FDA to "classify" a medical device?* Retrieved from http://www.fda.gov/AboutFDA/Transparency/Basics/ucm194438.htm.

FDA. (2016a). Deciding When to Submit a 510(k) for a Software Change to an Existing Device. In *Draft Guidance*: FDA,.

FDA. (2016b, 2016-11-07). Medical Device Reporting (MDR). Retrieved from https://www.fda.gov/MedicalDevices/safety/ReportaProblem/default.htm

FDA. (2016c). *Postmarket Management of Cybersecurity in Medical Devices*.

FDA. (2016d). Software As a Medical Device (SaMD): Clinical Evaluation. In *Draft Guidance*: FDA,.

FDA. (2016e). Use of Electronic Health Record Data in Clinical Investigations. In *Draft Guidance*: FDA,.

FDA. (2016f). *What is a medical device.* Retrieved from http://www.fda.gov/AboutFDA/Transparency/Basics/ucm211822.htm.

FDA. (2017a, 2017-03-08). Manufacturer and User Facility Device Experience Database - (MAUDE). Retrieved from https://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/PostmarketRequirements/ReportingAdverseEvents/ucm127891.htm

Medical Device Reporting, (2017b).

FDA. (2017c, 2017-01-03). What is a Medical Device Recall? Retrieved from https://www.fda.gov/MedicalDevices/Safety/ListofRecalls/ucm329946.htm

Fu, K. (2011). Trustworthy medical device software. *Public Health Effectiveness of the FDA, 510*, 102.

Galhotra, S., Amesur, N. B., Zajko, A. B., & Simmons, R. L. (2007). Migration of the Günther Tulip inferior vena cava filter to the chest. *Journal of Vascular and Interventional Radiology, 18*(12), 1581-1585.

Gardner, R. M., Clemmer, T. P., Evans, R. S., & Mark, R. G. (2014). Patient Monitoring Systems. In *Biomedical Informatics* (pp. 561-591): Springer.

Gunn, S. R. (1998). Support vector machines for classification and regression. *ISIS technical report, 14*, 85-86.

Halperin, D., Heydt-Benjamin, T. S., Ransford, B., Clark, S. S., Defend, B., Morgan, W., . . . Maisel, W. H. (2008). *Pacemakers and implantable cardiac defibrillators: Software radio attacks and zero-power defenses.* Paper presented at the Security and Privacy, 2008. SP 2008. IEEE Symposium on.

Hauser, R. G., & Kallinen, L. (2004). Deaths associated with implantable cardioverter defibrillator failure and deactivation reported in the United States Food and Drug Administration Manufacturer and User Facility Device Experience Database. *Heart Rhythm, 1*(4), 399-405.

Hebballi, N. B., Ramoni, R., Kalenderian, E., Delattre, V. F., Stewart, D. C., Kent, K., . . . Walji, M. F. (2015). The dangers of dental devices as reported in the food and drug administration manufacturer and user facility device experience database. *The Journal of the American Dental Association, 146*(2), 102-110.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics, 12*(1), 55-67.

Hwang, T. J., Sokolov, E., Franklin, J. M., & Kesselheim, A. S. (2016). Comparison of rates of safety issues and reporting of trial outcomes for medical devices approved in the European Union and United States: cohort study. *bmj, 353*, i3323.

Jiang, J. (2012). Information extraction from text. *Mining text data*, 11-41.

Jiang, Z., Abbas, H., Jang, K. J., & Mangharam, R. (2016). The Challenges of High-Confidence Medical Device Software. *Computer, 49*(1), 34-42.

Johnson, K., Jimison, H. B., & Mandl, K. D. (2014). Consumer health informatics and personal health records. In *Biomedical Informatics* (pp. 517-539): Springer.

Jones, R. G., Johnson, O. A., & Batstone, G. (2014). Informatics and the clinical laboratory. *The Clinical Biochemist Reviews, 35*(3), 177.

Karsh, B.-T., Weinger, M. B., Abbott, P. A., & Wears, R. L. (2010). Health information technology: fallacies and sober realities. *Journal of the American Medical Informatics Association, 17*(6), 617-623.

Kramer, D. B., Baker, M., Ransford, B., Molina-Markham, A., Stewart, Q., Fu, K., & Reynolds, M. R. (2012). Security and privacy qualities of medical devices: An analysis of FDA postmarket surveillance. *PLoS One, 7*(7), e40200.

Kwazneski, D., Six, C., & Stahlfeld, K. (2013). The unacknowledged incidence of laparoscopic stapler malfunction. *Surgical endoscopy, 27*(1), 86-89.

Lee, I., Pappas, G. J., Cleaveland, R., Hatcliff, J., Krogh, B. H., Lee, P., . . . Sha, L. (2006). High-confidence medical device software and systems. *Computer, 39*(4), 33-38.

Lewis, D. D. (1998). *Naive (Bayes) at forty: The independence assumption in information retrieval.* Paper presented at the European conference on machine learning.

Magrabi, F., Ong, M.-s., Runciman, W., & Coiera, E. (2011). *Patient safety problems associated with heathcare information technology: an analysis of adverse events reported to the US Food and Drug Administration.* Paper presented at the AMIA Annual Symposium Proceedings.

Magrabi, F., Ong, M.-S., Runciman, W., & Coiera, E. (2012). Using FDA reports to inform a classification for health information technology safety problems. *Journal of the American Medical Informatics Association, 19*(1), 45-53.

Manoucheri, E., Fuchs-Weizman, N., Cohen, S. L., Wang, K. C., & Einarsson, J. (2014). MAUDE: analysis of robotic-assisted gynecologic surgery. *Journal of minimally invasive gynecology, 21*(4), 592-595.

McDonald, C. J., Tang, P. C., & Hripcsak, G. (2014). Electronic health record systems. In *Biomedical Informatics* (pp. 391-421): Springer.

Meeks, D. W., Smith, M. W., Taylor, L., Sittig, D. F., Scott, J. M., & Singh, H. (2014). An analysis of electronic health record-related patient safety concerns. *Journal of the American Medical Informatics Association, 21*(6), 1053-1059.

Menon, S., Singh, H., Giardina, T. D., Rayburn, W. L., Davis, B. P., Russo, E. M., & Sittig, D. F. (2016). Safety huddles to proactively identify and address electronic health record safety. *Journal of the American Medical Informatics Association*, ocw153.

Merriam-Webster. (Ed.) (2017) Merriam-Webster.

Mintz, M., Bills, S., Snow, R., & Jurafsky, D. (2009). *Distant supervision for relation extraction without labeled data.* Paper presented at the Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2.

Mitchell, T. M. (1997). *Machine Learning*: McGraw-Hill, Inc.

Mosa, A. S. M., Yoo, I., & Sheets, L. (2012). A systematic review of healthcare applications for smartphones. *BMC medical informatics and decision making, 12*(1), 67.

Narula, G. (2017, 2017-09-14). Everyday Examples of Artificial Intelligence and Machine Learning. Retrieved from https://www.techemergence.com/everyday-examples-of-ai/

Naumann, R. W., & Brown, J. (2015). Complications of electromechanical morcellation reported in the manufacturer and user facility device experience (MAUDE) database. *Journal of minimally invasive gynecology, 22*(6), 1018-1021.

NCI. (2017, 2017-03-31). FDA Terminology.

Ng, A. Y., & Jordan, M. I. (2002). *On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes.* Paper presented at the Advances in neural information processing systems.

Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM. *Machine learning, 39*(2), 103-134.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Dubourg, V. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research, 12*(Oct), 2825-2830.

Rajan, P. V., Kramer, D. B., & Kesselheim, A. S. (2015). Medical Device Postapproval Safety Monitoring. *Circulation: Cardiovascular Quality and Outcomes, 8*(1), 124-131.

Rakitin, R. (2006). Coping with defective software in medical devices. *Computer, 39*(4), 40-45.

Resnic, F. S., & Normand, S.-L. T. (2012). Postmarketing surveillance of medical devices—filling in the gaps. *New England Journal of Medicine, 366*(10), 875-877.

Reynolds, I. S., Rising, J. P., Coukell, A. J., Paulson, K. H., & Redberg, R. F. (2014). Assessing the safety and effectiveness of devices after US Food and Drug Administration approval: FDA-mandated postapproval studies. *JAMA internal medicine, 174*(11), 1773-1779.

Rising, J., & Moscovitch, B. (2014). The Food and Drug Administration's unique device identification system: better postmarket data on the safety and effectiveness of medical devices. *JAMA internal medicine, 174*(11), 1719-1720.

Rubin, D. L., Greenspan, H., & Brinkley, J. F. (2014). Biomedical imaging informatics. In *Biomedical Informatics* (pp. 285-327): Springer.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature, 323*(6088), 533.

Samuel, A. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM journal of research and development, 3*(3), 210-229.

Sandler, K., Ohrstrom, L., Moy, L., & McVay, R. (2010). Killed by code: Software transparency in implantable medical devices. *Software Freedom Law Center*, 308-319.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR), 34*(1), 1-47.

Sfyroeras, G. S., Koutsiaris, A., Karathanos, C., Giannakopoulos, A., & Giannoukas, A. D. (2010). Clinical relevance and treatment of carotid stent fractures. *Journal of vascular surgery, 51*(5), 1280-1285.

Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., . . . Lunt, B. (2006). *Computing curricula 2005: The overview report.* Paper presented at the ACM SIGCSE Bulletin.

Simon, H. A. (1983). Why should machines learn? In *Machine learning* (pp. 25-37): Springer.

Starren, J. B., Nesbitt, T. S., & Chiang, M. F. (2014). Telehealth. In *Biomedical Informatics* (pp. 541-560): Springer.

Taylor, R. H., Menciassi, A., Fichtinger, G., & Dario, P. (2008). Medical robotics and computer-integrated surgery. In *Springer handbook of robotics* (pp. 1199-1222): Springer.

Thirumalai, S., & Sinha, K. K. (2011). Product recalls in the medical device industry: An empirical exploration of the sources and financial consequences. *Management Science, 57*(2), 376-392.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267-288.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind, 59*(236), 433-460.

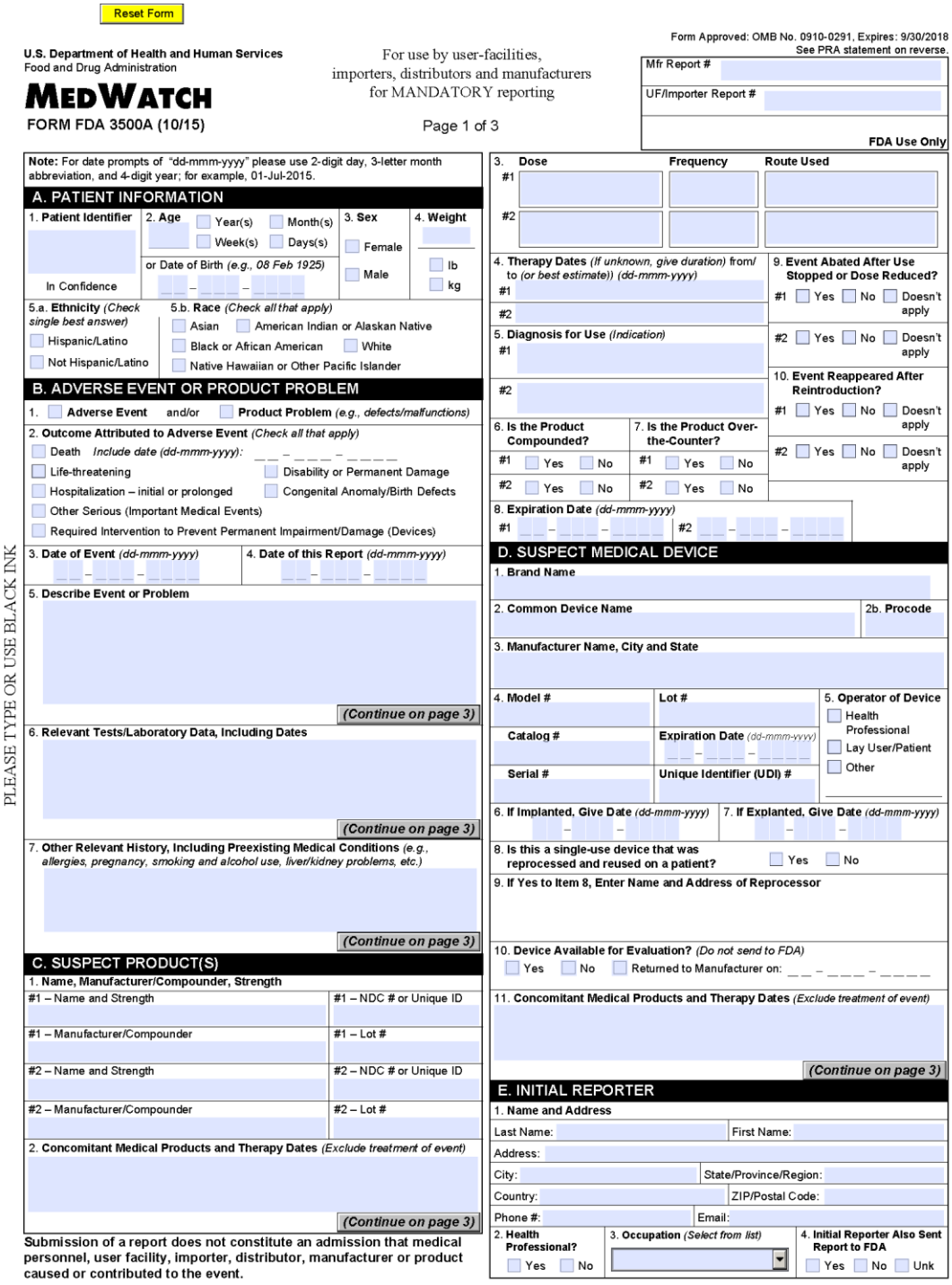
Wallace, D. R., & Kuhn, D. R. (2001). Failure modes in medical device software: an analysis of 15 years of recall data. *International Journal of Reliability, Quality and Safety Engineering, 8*(04), 351-371.

Ward, J. R., & Clarkson, P. J. (2004). An analysis of medical device-related errors: prevalence and possible solutions. *Journal of medical engineering & technology, 28*(1), 2-21.

Woodcock, J., Larsen, P. G., Bicarregui, J., & Fitzgerald, J. (2009). Formal methods: Practice and experience. *ACM Computing Surveys (CSUR), 41*(4), 19.

Zuckerman, D. M., Brown, P., & Nissen, S. E. (2011). Medical device recalls and the FDA approval process. *Archives of internal medicine, 171*(11), 1006-1011.

APPENDIX A: MEDWATCH FORM FDA 3500A



Reset Form

U.S. Department of Health and Human Services
Food and Drug Administration

MEDWATCH
FORM FDA 3500A (10/15)

For use by user-facilities, importers, distributors and manufacturers for MANDATORY reporting

Page 1 of 3

Form Approved: OMB No. 0910-0291, Expires: 9/30/2018
See PRA statement on reverse.

Mfr Report #
UF/Importer Report #
FDA Use Only

PLEASE TYPE OR USE BLACK INK

Note: For date prompts of "dd-mmm-yyyy" please use 2-digit day, 3-letter month abbreviation, and 4-digit year; for example, 01-Jul-2015.

A. PATIENT INFORMATION

1. Patient Identifier — In Confidence
2. Age — Year(s) Month(s) Week(s) Days(s); or Date of Birth (e.g., 08 Feb 1925)
3. Sex — Female Male
4. Weight — lb kg
5.a. Ethnicity (Check single best answer) — Hispanic/Latino; Not Hispanic/Latino
5.b. Race (Check all that apply) — Asian; American Indian or Alaskan Native; Black or African American; White; Native Hawaiian or Other Pacific Islander

B. ADVERSE EVENT OR PRODUCT PROBLEM

1. Adverse Event and/or Product Problem (e.g., defects/malfunctions)
2. Outcome Attributed to Adverse Event (Check all that apply)
Death Include date (dd-mmm-yyyy): ___ – ____ – _____
Life-threatening; Disability or Permanent Damage; Hospitalization – initial or prolonged; Congenital Anomaly/Birth Defects; Other Serious (Important Medical Events); Required Intervention to Prevent Permanent Impairment/Damage (Devices)
3. Date of Event (dd-mmm-yyyy)
4. Date of this Report (dd-mmm-yyyy)
5. Describe Event or Problem
(Continue on page 3)
6. Relevant Tests/Laboratory Data, Including Dates
(Continue on page 3)
7. Other Relevant History, Including Preexisting Medical Conditions (e.g., allergies, pregnancy, smoking and alcohol use, liver/kidney problems, etc.)
(Continue on page 3)

C. SUSPECT PRODUCT(S)

1. Name, Manufacturer/Compounder, Strength

#1 – Name and Strength	#1 – NDC # or Unique ID
#1 – Manufacturer/Compounder	#1 – Lot #
#2 – Name and Strength	#2 – NDC # or Unique ID
#2 – Manufacturer/Compounder	#2 – Lot #

2. Concomitant Medical Products and Therapy Dates (Exclude treatment of event)
(Continue on page 3)

Submission of a report does not constitute an admission that medical personnel, user facility, importer, distributor, manufacturer or product caused or contributed to the event.

3. Dose / Frequency / Route Used
#1
#2
4. Therapy Dates (If unknown, give duration) from/to (or best estimate)) (dd-mmm-yyyy)
#1
#2
5. Diagnosis for Use (Indication)
#1
#2
6. Is the Product Compounded? #1 Yes No; #2 Yes No
7. Is the Product Over-the-Counter? #1 Yes No; #2 Yes No
8. Expiration Date (dd-mmm-yyyy) #1 ___ – ____ – _____ #2 ___ – ____ – _____
9. Event Abated After Use Stopped or Dose Reduced? #1 Yes No Doesn't apply; #2 Yes No Doesn't apply
10. Event Reappeared After Reintroduction? #1 Yes No Doesn't apply; #2 Yes No Doesn't apply

D. SUSPECT MEDICAL DEVICE

1. Brand Name
2. Common Device Name
2b. Procode
3. Manufacturer Name, City and State
4. Model # / Lot # / Catalog # / Expiration Date (dd-mmm-yyyy) / Serial # / Unique Identifier (UDI) #
5. Operator of Device — Health Professional; Lay User/Patient; Other
6. If Implanted, Give Date (dd-mmm-yyyy)
7. If Explanted, Give Date (dd-mmm-yyyy)
8. Is this a single-use device that was reprocessed and reused on a patient? Yes No
9. If Yes to Item 8, Enter Name and Address of Reprocessor
10. Device Available for Evaluation? (Do not send to FDA) Yes No Returned to Manufacturer on: ___ – ____ – _____
11. Concomitant Medical Products and Therapy Dates (Exclude treatment of event)
(Continue on page 3)

E. INITIAL REPORTER

1. Name and Address
Last Name: First Name:
Address:
City: State/Province/Region:
Country: ZIP/Postal Code:
Phone #: Email:
2. Health Professional? Yes No
3. Occupation (Select from list)
4. Initial Reporter Also Sent Report to FDA Yes No Unk

Reset Form

MEDWATCH

FORM FDA 3500A (10/15) *(continued)*

FDA USE ONLY

F. FOR USE BY USER FACILITY/IMPORTER *(Devices Only)*

1. **Check One** ☐ User Facility ☐ Importer
2. **UF/Importer Report Number**
3. **User Facility or Importer Name/Address**
4. **Contact Person**
5. **Phone Number**
6. **Date User Facility or Importer Became Aware of Event** *(dd-mmm-yyyy)*
7. **Type of Report** ☐ Initial ☐ Follow-up # _____
8. **Date of This Report** *(dd-mmm-yyyy)*
9. **Approximate Age of Device**
10. **Event Problem Codes** *(Refer to coding manual)* Patient Code – – Device Code – –
11. **Report Sent to FDA?** *(If Yes, enter date (dd-mmm-yyyy))* ☐ Yes ☐ No
12. **Location Where Event Occurred** ☐ Hospital ☐ Home ☐ Nursing Home ☐ Outpatient Treatment Facility ☐ Outpatient Diagnostic Facility ☐ Ambulatory Surgical Facility ☐ Other: _____ *(Specify)*
13. **Report Sent to Manufacturer?** *(If Yes, enter date (dd-mmm-yyyy))* ☐ Yes ☐ No
14. **Manufacturer Name/Address**

G. ALL MANUFACTURERS

1. **Contact Office (and Manufacturing Site for Devices)** Name / Address / Email Address / Compounding Outsourcing Facility 503B? ☐ Yes
2. **Phone Number**
3. **Report Source** *(Check all that apply)* ☐ Foreign ☐ Study ☐ Literature ☐ Consumer ☐ Health Professional ☐ User Facility ☐ Company Representative ☐ Distributor ☐ Other:
4. **Date Received by Manufacturer** *(dd-mmm-yyyy)*
5. NDA # / ANDA # / IND # / BLA # / PMA/510(k) # / Combination Product ☐ Yes / Pre-1938 ☐ Yes / OTC ☐ Yes
6. **If IND, Give Protocol #**
7. **Type of Report** *(Check all that apply)* ☐ 5-day ☐ 7-day ☐ 10-day ☐ 15-day ☐ 30-day ☐ Periodic ☐ Initial ☐ Follow-up # ____
8. **Adverse Event Term(s)**
9. **Manufacturer Report Number**

H. DEVICE MANUFACTURERS ONLY

1. **Type of Reportable Event** ☐ Death ☐ Serious Injury ☐ Malfunction
2. **If Follow-up, What Type?** ☐ Correction ☐ Additional Information ☐ Response to FDA Request ☐ Device Evaluation
3. **Device Evaluated by Manufacturer?** ☐ Not Returned to Manufacturer ☐ Yes ☐ Evaluation Summary Attached ☐ No *(Attach page to explain why not)* or provide code:
4. **Device Manufacture Date** *(dd-mmm-yyyy)*
5. **Labeled for Single Use?** ☐ Yes ☐ No
6. **Event Problem and Evaluation Codes** *(Refer to coding manual)* Patient Code – – Device Code – – Method – – – – Results – – – – Conclusions – – – –
7. **If Remedial Action Initiated, Check Type** ☐ Recall ☐ Repair ☐ Replace ☐ Relabeling ☐ Notification ☐ Inspection ☐ Patient Monitoring ☐ Modification/Adjustment ☐ Other:
8. **Usage of Device** ☐ Initial Use of Device ☐ Reuse ☐ Unknown
9. **If action reported to FDA under 21 USC 360i(f), list correction/removal reporting number:**
10. ☐ **Additional Manufacturer Narrative** and / or 11. ☐ **Corrected Data**

This section applies only to requirements of the Paperwork Reduction Act of 1995.

The public reporting burden for this collection of information has been estimated to average 73 minutes per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to:

Department of Health and Human Services
Food and Drug Administration
Office of Chief Information Officer
Paperwork Reduction Act (PRA) Staff
PRAStaff@fda.hhs.gov

Please DO NOT RETURN this form to the above PRA Staff email address.

OMB Statement: "An agency may not conduct or sponsor, and a person is not required to respond to, a collection of information unless it displays a currently valid OMB control number."

Reset Form | Delete Page

(CONTINUATION PAGE)
For use by user-facilities,
importers, distributors, and manufacturers
for MANDATORY reporting

MEDWATCH
FORM FDA 3500A (10/15) *(continued)*

Back to Item B.5

B.5. **Describe Event or Problem** *(continued)*

Back to Item B.6

B.6. **Relevant Tests/Laboratory Data, Including Dates** *(continued)*

Back to Item B.7

B.7. **Other Relevant History, Including Preexisting Medical Conditions** *(e.g., allergies, pregnancy, smoking and alcohol use, hepatic/renal dysfunction, etc.) (continued)*

Back to Item C.2

Concomitant Medical Products and Therapy Dates *(Exclude treatment of event) (For continuation of C.2 and/or D.11; please distinguish)*

Back to Item D.11

Other Remarks

APPENDIX B: MEDWATCH FORM FDA 3500

Reset Form

U.S. Department of Health and Human Services

MEDWATCH

The FDA Safety Information and Adverse Event Reporting Program

For VOLUNTARY reporting of adverse events, product problems and product use errors

Page 1 of 3

Form Approved: OMB No. 0910-0291, Expires: 9/30/2018
See PRA statement on reverse.

FDA USE ONLY

Triage unit sequence #

FDA Rec. Date

PLEASE TYPE OR USE BLACK INK

Note: For date prompts of "dd-mmm-yyyy" please use 2-digit day, 3-letter month abbreviation, and 4-digit year; for example, 01-Jul-2015.

A. PATIENT INFORMATION

1. **Patient Identifier** — In Confidence

2. **Age** ☐ Year(s) ☐ Month(s) ☐ Week(s) ☐ Days(s)
or Date of Birth *(e.g., 08 Feb 1925)* __ – ___ – ____

3. **Sex** ☐ Female ☐ Male

4. **Weight** ☐ lb ☐ kg

5.a. **Ethnicity** *(Check single best answer)*
☐ Hispanic/Latino
☐ Not Hispanic/Latino

5.b. **Race** *(Check all that apply)*
☐ Asian ☐ American Indian or Alaskan Native
☐ Black or African American ☐ White
☐ Native Hawaiian or Other Pacific Islander

B. ADVERSE EVENT, PRODUCT PROBLEM

1. **Check all that apply**
☐ **Adverse Event** ☐ **Product Problem** *(e.g., defects/malfunctions)*
☐ **Product Use Error** ☐ **Problem with Different Manufacturer of Same Medicine**

2. **Outcome Attributed to Adverse Event** *(Check all that apply)*
☐ Death *Include date (dd-mmm-yyyy):* __ – ___ – ____
☐ Life-threatening ☐ Disability or Permanent Damage
☐ Hospitalization – initial or prolonged ☐ Congenital Anomaly/Birth Defects
☐ Other Serious (Important Medical Events)
☐ Required Intervention to Prevent Permanent Impairment/Damage (Devices)

3. **Date of Event** *(dd-mmm-yyyy)* __ – ___ – ____

4. **Date of this Report** *(dd-mmm-yyyy)* __ – ___ – ____

5. **Describe Event, Problem or Product Use Error**

(Continue on page 3)

6. **Relevant Tests/Laboratory Data, Including Dates**

(Continue on page 3)

7. **Other Relevant History, Including Preexisting Medical Conditions** *(e.g., allergies, pregnancy, smoking and alcohol use, liver/kidney problems, etc.)*

(Continue on page 3)

C. PRODUCT AVAILABILITY

2. **Product Available for Evaluation?** *(Do not send product to FDA)*
☐ Yes ☐ No ☐ Returned to Manufacturer on *(dd-mmm-yyyy)* __ – ___ – ____

D. SUSPECT PRODUCTS

1. **Name, Manufacturer/Compounder, Strength** *(from product label)*

#1 – Name and Strength	#1 – NDC # or Unique ID
#1 – Manufacturer/Compounder	#1 – Lot #
#2 – Name and Strength	#2 – NDC # or Unique ID
#2 – Manufacturer/Compounder	#2 – Lot #

3. **Dose or Amount** / **Frequency** / **Route**
#1
#2

4. **Dates of Use** *(From/To for each) (If unknown, give duration, or best estimate) (dd-mmm-yyyy)*
#1
#2

5. **Diagnosis or Reason for Use** *(indication)*
#1
#2

6. **Is the Product Compounded?**
#1 ☐ Yes ☐ No
#2 ☐ Yes ☐ No

7. **Is the Product Over-the-Counter?**
#1 ☐ Yes ☐ No
#2 ☐ Yes ☐ No

8. **Expiration Date** *(dd-mmm-yyyy)*
#1 __ – ___ – ____ #2 __ – ___ – ____

9. **Event Abated After Use Stopped or Dose Reduced?**
#1 ☐ Yes ☐ No ☐ Doesn't apply
#2 ☐ Yes ☐ No ☐ Doesn't apply

10. **Event Reappeared After Reintroduction?**
#1 ☐ Yes ☐ No ☐ Doesn't apply
#2 ☐ Yes ☐ No ☐ Doesn't apply

E. SUSPECT MEDICAL DEVICE

1. **Brand Name**

2. **Common Device Name** / 2b. **Procode**

3. **Manufacturer Name, City and State**

4. **Model #** / **Lot #** / 5. **Operator of Device** ☐ Health Professional ☐ Lay User/Patient ☐ Other

Catalog # / **Expiration Date** *(dd-mmm-yyyy)* __ – ___ – ____

Serial # / **Unique Identifier (UDI) #**

6. **If Implanted. Give Date** *(dd-mmm-yyyy)* __ – ___ – ____

7. **If Explanted. Give Date** *(dd-mmm-yyyy)* __ – ___ – ____

8. **Is this a single-use device that was reprocessed and reused on a patient?** ☐ Yes ☐ No

9. **If Yes to Item 8, Enter Name and Address of Reprocessor**

F. OTHER (CONCOMITANT) MEDICAL PRODUCTS

Product names and therapy dates *(Exclude treatment of event)*

(Continue on page 3)

G. REPORTER *(See confidentiality section on back)*

1. **Name and Address**

Last Name: / First Name:
Address:
City: / State/Province/Region:
Country: / ZIP/Postal Code:
Phone #: / Email:

2. **Health Professional?** ☐ Yes ☐ No

3. **Occupation**

4. **Also Reported to:** ☐ Manufacturer/Compounder ☐ User Facility ☐ Distributor/Importer

5. **If you do NOT want your identity disclosed to the manufacturer, please mark this box:** ☐

FORM FDA 3500 (10/15) Submission of a report does not constitute an admission that medical personnel or the product caused or contributed to the event.

ADVICE ABOUT VOLUNTARY REPORTING

Detailed instructions available at: http://www.fda.gov/medwatch/report/consumer/instruct.htm

Report adverse events, product problems or product use errors with:

- Medications *(drugs or biologics)*
- Medical devices *(including in-vitro diagnostics)*
- Combination products *(medication & medical devices)*
- Human cells, tissues, and cellular and tissue-based products
- Special nutritional products *(dietary supplements, medical foods, infant formulas)*
- Cosmetics
- Food *(including beverages and ingredients added to foods)*

Report product problems - quality, performance or safety concerns such as:

- Suspected counterfeit product
- Suspected contamination
- Questionable stability
- Defective components
- Poor packaging or labeling
- Therapeutic failures (product didn't work)

-Fold Here-

Report SERIOUS adverse events. An event is serious when the patient outcome is:

- Death
- Life-threatening
- Hospitalization - initial or prolonged
- Disability or permanent damage
- Congenital anomaly/birth defect
- Required intervention to prevent permanent impairment or damage (devices)
- Other serious (important medical events)

Report even if:

- You're not certain the product caused the event
- You don't have all the details

How to report:

- Just fill in the sections that apply to your report
- Use section D for all products except medical devices
- Attach additional pages if needed
- Use a separate form for each patient
- Report either to FDA or the manufacturer *(or both)*

Other methods of reporting:

- 1-800-FDA-0178 - To FAX report
- 1-800-FDA-1088 - To report by phone
- www.fda.gov/medwatch/report.htm - To report online

If your report involves a serious adverse event with a device and it occurred in a facility outside a doctor's office, that facility may be legally required to report to FDA and/or the manufacturer. Please notify the person in that facility who would handle such reporting.

-Fold Here-

If your report involves a serious adverse event with a vaccine, call 1-800-822-7967 to report.

Confidentiality: The patient's identity is held in strict confidence by FDA and protected to the fullest extent of the law. The reporter's identity, including the identity of a self-reporter, may be shared with the manufacturer unless requested otherwise.

The information in this box applies only to requirements of the Paperwork Reduction Act of 1995

The burden time for this collection of information has been estimated to average 40 minutes per response, including the time to review instructions, search existing data sources, gather and maintain the data needed, and complete and review the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to:

Department of Health and Human Services *Food and Drug Administration* *Office of Chief Information Officer* *Paperwork Reduction Act (PRA) Staff* *PRAStaff@fda.hhs.gov*	*Please DO NOT RETURN this form to the PRA Staff e-mail to the left.*	*OMB statement:* *"An agency may not conduct or sponsor, and a person is not required to respond to, a collection of information unless it displays a currently valid OMB control number."*

U.S. DEPARTMENT OF HEALTH AND HUMAN SERVICES
Food and Drug Administration

FORM FDA 3500 (10/15) (Back) **Please Use Address Provided Below -- Fold in Thirds, Tape and Mail**

DEPARTMENT OF HEALTH & HUMAN SERVICES

Public Health Service
Food and Drug Administration
Rockville, MD 20857

Official Business
Penalty for Private Use $300

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES OR APO/FPO

BUSINESS REPLY MAIL
FIRST CLASS MAIL PERMIT NO. 946 ROCKVILLE MD

POSTAGE WILL BE PAID BY FOOD AND DRUG ADMINISTRATION

MEDWATCH
The FDA Safety Information and Adverse Event Reporting Program
Food and Drug Administration
5600 Fishers Lane
Rockville, MD 20852-9787

Reset Form

U.S. Department of Health and Human Services

MEDWATCH

The FDA Safety Information and Adverse Event Reporting Program

FORM FDA 3500 (10/15) *(continued)*

(CONTINUATION PAGE)

For VOLUNTARY reporting of adverse events and product problems

Back to Form

B.5. **Describe Event or Problem** *(continued)*

Back to Form

B.6. **Relevant Tests/Laboratory Data, Including Dates** *(continued)*

Back to Form

B.7. **Other Relevant History, Including Preexisting Medical Conditions** *(e.g., allergies, pregnancy, smoking and alcohol use, hepatic/renal dysfunction, etc.) (continued)*

Back to Form

F. Concomitant Medical Products and Therapy Dates *(Exclude treatment of event) (continued)*

APPENDIX C: DEVICE PROBLEM CODE HIERARCHY

Presented below, is an excerpt from the FDA's Device Problem Code Hierarchy (DPCH) listing all computer software related problems:

- **Computer Software Issue C63269; FDA 1112** – *Issue associated with written programs, codes, and/or software system that affects device performance or communication with another device.*
  - **Application Network Issue C64349; FDA 2879** – *Issue associated with the deviations from documented system specifications that affects overall system performance and/or the performance of an individual device or collection of devices connected to that system.*
  - **Application Program Issue C63305; FDA 2880** – *Issue associated with the requirement for software to fulfill its function within an intended use or application.*
    - **Application Interface becomes nonfunctional or program exits abnormally C63306; FDA 1138**
    - **Application Program Version or Upgrade Problem C63304; FDA 2881**
    - **Incorrect Error Code C63206; FDA 2963**
    - **Problem with Software Installation C67507; FDA 3013** – *Issue associated with installing the device software in a manner that allows full functioning of the device. Source of installation could be manufacturer or user.*
    - **Programming Issue C62839; FDA 3014** – *Issue associated with the written program code or application software used to satisfy a stated need or objective for functioning of the device. These do not include issues associated with the operating system*
      - **Incorrect Software Programming Calculations C63081; FDA 1495**
        - **Dose Calculation Error due to Software Problem C63220; FDA 1189**
        - **Parameter Calculation Error due to Software Problem C63083; FDA 1449**
        - **Power Calculation Error due to Software Problem C62916; FDA 1473**
      - **Medication Error C91396; FDA 3198** - *Event in which the device software design results in errors of medication preparation or administration.*
  - **Data Issue C91397; FDA 3196** - *Event in which data (charting, orders, results) is not correctly stored, transferred, updated, or displayed.*
    - **Loss of Data C63257; FDA 2903** - *Event in which information is unintentionally permanently or temporarily lost, deleted, corrupted, or overwritten.*
    - **Patient Data Issue C91398; FDA 3197** - *Event in which information is accessed by the healthcare provider and either the wrong patient or the wrong information is retrieved despite correct inquiry procedures.*
  - **Computer Operating System Issue C63270; FDA 2898** – *Issue associated with the machinery operating system, a collection of software, firmware, and hardware elements that control the execution of computer programs and provides such services as computer resource allocation, job control, input/output control, and file management in a computer system.*
    - **Operating System Becomes Nonfunctional C62894; FDA 2996** – *Issue associated with malfunction of the computer operating system as opposed to an application software issue.*
    - **Operating System Version or Upgrade Problem C62893; FDA 2997** – *Issue associated with replacing an older operation system to an up-to-date operation system.*
  - **Computer System Security Issue C64348; FDA 2899**
    - **Unauthorized Access to Computer System C63259; FDA 3025** – *Issue associated with an access that was not permitted to the computer system that may lead to modification of program, corruption of data, or and break in network security. This concept is closely associated with computer integrity which is the degree to which a system or component prevents unauthorized access to, or modification of, computer programs or data.*
      - **Application Security Issue C63042; FDA 2882** – *Issue associated with the acquisition of computer programming codes that can replicate and spread from one computer system to another thereby leading to damaged software, hardware and data.*
  - **Data Back-Up Problem C63258; FDA 2902** – *Issue associated with problems relating to a system, component, file, procedure, or person available to replace or help restore a primary item in the event of a failure or externally caused disaster.*
    - **Failure to Back-Up C63195; FDA 1047** – *Issue associated with the inability to backup or to retrieve a backed up version (corrupted file) of device data or system files.*
    - **Failure to Convert to Back-Up C63189; FDA 1048**
  - **Date-related software issue C67508; FDA 2582** – *Issue associated with programming of calendar dates and/or time as a factor in the operation of a medical device.*

APPENDIX D: POSITIVE PATTERNS FOR SEED DATA

Candidate records for positive seed data from the corpus were identified using a set of regular expression-based patterns. Each pattern was a concatenation of a 'main' term with a prefix or postfix/suffix term:

$$pattern_1 = prefix\_term + ' ' + main\_term$$

$$pattern_2 = main\_term + ' ' + postfix\_term$$

In the tables below, we list all values for the main terms, prefix terms and postfix terms used in building the collection of patterns:

Main Terms

ALGORITHM
BLACK SCREEN
BLACKSCREEN
BLANK SCREEN
BLANKSCREEN
BLUE SCREEN
BLUESCREEN
BOOT UP
BOOT
BOOTSTRAP
BOOTUP
BUFFER OVER FLOW

BUFFER OVERFLOW
BUFFER OVER- FLOW
BUFFEROVERFLOW
C\+\+
CALCULATION
CHECKSUM
COBOL
COMM
COMMAND
COMMUNICATION
COMMUNICATIONS

COMPUTATION
COMPUTE
COMPUTER
CONNECTION
CONNECTIVITY
CPOE
CPU
DATA ACQUISITION
DATA ANALYSIS
DATA BASE
DATA CAPTURE
DATA LOSS

DATA MANAGEMENT SYSTEM
DATA MANAGEMENT
DATA PROCESSING
DATA STORAGE
DATA TRANSFER
DATABASE
DEBUG
DECISION SUPPORT SYSTEM
DECODE
DECODING
DEVICE DRIVER
DICOM
DISK DRIVE
DOWNLOAD
DVI CABLE
DVI
HER
ELECTRONIC HEALTH RECORD

ELECTRONIC MEDICAL RECORD
EMAIL
E-MAIL
EMR
ENCODE
ENCODING
ETHERNET CABLE
ETHERNET
FILE SYSTEM
FILE TRANSFER
FILE
FIRM WARE
FIRMWARE
FIRM-WARE
FLASH CARD
FLASH MEMORY
FPGA
FTP
GRAPHICAL USER INTERFACE
GRAPHICS
GRAPHING

GUI
HARD DISK
HARD DRIVE
HASH
HDMI CABLE
HDMI
HIT
HL7
HTTP
IBM
IMAGE DISPLAY
IMAGE RECOGNITION
IMAGING SYSTEM
INFORMATION SYSTEM
INFORMATION TECHNOLOGY
INTEL
INTERNET
INTEROPERABILITY
INTRANET
IT SYSTEM

JAVA
JAVASCRIPT
JSON
KEY BOARD
KEYBOARD
KEYPAD
LABORATORY INFORMATION SYSTEM
LAPTOP
LIMS
LINUX
LIS
LOGIC
LOGIN
MAINFRAME
MEMORY OVERFLOW
MEMORY OVERWRITE
MESSAGING
MICRO CONTROLLER

MICROCONTROLLER
MICROSOFT
MICROSYSTEM
MIDDLE WARE
MIDDLEWARE
MODEM
MOTHER BOARD
MOTHERBOARD
MOUSE
NETWORK
NETWORKING
OPERATING SYSTEM
ORACLE
PCMCIA
PING
PLOTTING
PRINTING
PROCESSOR
PROGRAMMER
PROGRAMMING
PYTHON
QUERY

RADIOLOGY INFORMATION SYSTEM
REMOTE SYSTEM
ROBOT
ROBOTIC
ROBOTICS
RULE ENGINE
RULE
RULES ENGINE
RUN TIME
RUNTIME
RUN-TIME
SCREEN
SCRIPT
SCRIPTING
SERVER
SETTINGS
SMART CARD
SMARTCARD
SOAP
SOFT WARE
SOFTWARE SYSTEM

SOFTWARE UPDATE
SOFTWARE UPGRADE
SOFTWARE
SOFT-WARE
SOURCE CODE
SQL
SYNC
SYNCH
SYNCHRONIZATION
TCP
TCP/IP
TELNET
TOUCH SCREEN
TOUCHSCREEN
TRASNSMISSION
UNIX
UPLOAD
USER INTERFACE
UUDI
VBSCRIPT
VGA CABLE
VGA

VOICE RECOGNITION
WEB SITE
WEBSITE
WHITE SCREEN
WHITESCREEN
WINDOWS
WSDL
XML
Y2K

Prefix Terms

CORRUPT
CORRUPTED
DAMAGED
DEFECTIVE
ERRONEOUS RESULT.*
ERROR (\w+\s)?CODE.*
ERROR (\w+\s)?MESSAGE.*
FAILED

FAULTY
INCORRECT RESULT.*
UNSTABLE

Postfix Terms:

.*ERRONEOUS RESULT
.*ERROR (\w+\s)?CODE
.*ERROR (\w+\s)?MESSAGE
.*INCORRECT RESULT
ANOMALY
BUG
CORRUPT
CRASH
DAMAGE
DEFECT
ERROR
EXCEPTION
FAIL

FAULT
FREEZE
FROZE

HANG
HUNG
ISSUE

MALFUNCTION
PROBLEM

APPENDIX E: STOP WORDS

Through a series of experimental classification runs, we discovered several terms that were highly common but not discriminative. We excluded 569 different terms from the list of features. These included 318 commonly occurring English words defined by Glasgow Information Retrieval Group and materialized (except the word 'computer') in the set of stop words in Python's scikit-learn module (Pedregosa et al., 2011), and 251 terms we identified as noise through our own experiments. The following tables contain these stop words.

Scikit-Learn English Stop-words

A	ALSO	ANYONE
ABOUT	ALTHOUGH	ANYTHING
ABOVE	ALWAYS	ANYWAY
ACROSS	AM	ANYWHERE
AFTER	AMONG	ARE
AFTERWARDS	AMONGST	AROUND
AGAIN	AMOUNGST	AS
AGAINST	AMOUNT	AT
ALL	AN	BACK
ALMOST	AND	BE
ALONE	ANOTHER	BECAME
ALONG	ANY	BECAUSE
ALREADY	ANYHOW	BECOME

BECOMES
BECOMING
BEEN
BEFORE
BEFOREHAND
BEHIND
BEING
BELOW
BESIDE
BESIDES
BETWEEN
BEYOND
BILL
BOTH
BOTTOM
BUT
BY
CALL
CAN
CANNOT
CANT
CO
CON

COULD
COULDNT
CRY
DE
DESCRIBE
DETAIL
DO
DONE
DOWN
DUE
DURING
EACH
EG
EIGHT
EITHER
ELEVEN
ELSE
ELSEWHERE
EMPTY
ENOUGH
ETC
EVEN
EVER

EVERY
EVERYONE
EVERYTHING
EVERYWHERE
EXCEPT
FEW
FIFTEEN
FIFTY
FILL
FIND
FIRE
FIRST
FIVE
FOR
FORMER
FORMERLY
FORTY
FOUND
FOUR
FROM
FRONT
FULL
FURTHER

GET
GIVE
GO
HAD
HAS
HASN'T
HAVE
HE
HENCE
HER
HERE
HEREAFTER
HEREBY
HEREIN
HEREUPON
HERS
HERSELF
HIM
HIMSELF
HIS
HOW
HOWEVER
HUNDRED

I
IE
IF
IN
INC
INDEED
INTEREST
INTO
IS
IT
ITS
ITSELF
KEEP
LAST
LATTER
LATTERLY
LEAST
LESS
LTD
MADE
MANY
MAY
ME

MEANWHILE
MIGHT
MILL
MINE
MORE
MOREOVER
MOST
MOSTLY
MOVE
MUCH
MUST
MY
MYSELF
NAME
NAMELY
NEITHER
NEVER
NEVERTHELESS
NEXT
NINE
NO
NOBODY
NONE

NOONE
NOR
NOT
NOTHING
NOW
NOWHERE
OF
OFF
OFTEN
ON
ONCE
ONE
ONLY
ONTO
OR
OTHER
OTHERS
OTHERWISE
OUR
OURS
OURSELVES
OUT
OVER

OWN
PART
PER
PERHAPS
PLEASE
PUT
RATHER
RE
SAME
SEE
SEEM
SEEMED
SEEMING
SEEMS
SERIOUS
SEVERAL
SHE
SHOULD
SHOW
SIDE
SINCE
SINCERE
SIX

SIXTY
SO
SOME
SOMEHOW
SOMEONE
SOMETHING
SOMETIME
SOMETIMES
SOMEWHERE
STILL
SUCH
SYSTEM
TAKE
TEN
THAN
THAT
THE
THEIR
THEM
THEMSELVES
THEN
THENCE
THERE

THEREAFTER
THEREBY
THEREFORE
THEREIN
THEREUPON
THESE
THEY
THICK
THIN
THIRD
THIS
THOSE
THOUGH
THREE
THROUGH
THROUGHOUT
THRU
THUS
TO
TOGETHER
TOO
TOP
TOWARD

TOWARDS
TWELVE
TWENTY
TWO
UN
UNDER
UNTIL
UP
UPON
US
VERY
VIA
WAS
WE
WELL
WERE
WHAT
WHATEVER
WHEN
WHENCE
WHENEVER
WHERE
WHEREAFTER

WHEREAS
WHEREBY
WHEREIN
WHEREUPON
WHEREVER
WHETHER
WHICH
WHILE
WHITHER
WHO
WHOEVER
WHOLE
WHOM
WHOSE
WHY
WILL
WITH
WITHIN
WITHOUT
WOULD
YET
YOU
YOUR

YOURS	YOURSELF	YOURSELVES

Custom Stop Words:

00	120	2007
000	121	2008
0000	1225714	2009
01	13	2010
02	14	2011
03	15	2012
04	16	2013
05	1627487	2014
06	17	2015
064	18	2016
069	19	21
07	20	2134265
08	200	22
09	2000	2210968
10	2002	2240
100	2003	23
105	2004	2367
11	2005	24
12	2006	25

250
26
27
28
29
30
300
31
320
322
33
35
3500A
36
3889
40
400
45
48
50
500
5076
510

510K
52
56
58
5MM
60
600
65
70
75
80
803
810
81011
812
814
840
8709
8709SC
90
92
95
9600

9800
9900
ABLE
ADDITIONAL
ADVANTAGE
ADVERSE
ADVISED
ALLEGING
ALTERNATE
ANIMAS
AVAILABLE
BASAL
BASED
BAXTER
BG
BIOMED
BLOOD
CALLED
CAUSED
CGM
CLOSED
COLLEAGUE
COLLIMATOR

COMPANY	EVALUATED	INVESTIGATION
COMPLAINT	EXPERIENCED	INVOLVEMENT
CONDUCTED	FACILITY	KNOWN
CONFIRMED	FAILED	LAY
CONTACT	FAILURE	LIFESCAN
CONTACTED	FINDINGS	LIP
CONTACTS	FOLLOW	LONGER
CONTAMINATION	FOLLOWING	MALFUNCTION
CONTRIBUTED	FSE	MATTER
COVIDIEN	GAS	MDR
CRACKED	GE	MEDTRONIC
CRACKS	GLUCOSE	MG
CUSTOMER	HARM	MISTREATMENT
DEATH	HOSP	MMOL
DETERMINED	HOSPITAL	MOTHER
DEXCOM	IDENTIFIED	NEED
DID	INCIDENT	NEW
DISCONTINUE	INFORMED	NOTED
DL	INFUSION	OBTAINS
DOES	INJURY	OCCLUSION
DOOR	INSULIN	OEC
DROOP	INTENDED	ONSITE
DWELL	INTERVENTION	OPERATES

ORTHO	READINGS	SPECTRUM
PATIENT	RECEIVED	STAFF
PATIENTS	RELATED	STATES
PERFORM	REMOVED	STRIP
PERFORMED	REP	SUBMISSION
PERSONNEL	REPLACED	SVC
PHILIPS	REPORTED	TESTING
PLACED	REPORTEDLY	THERAPY
PLUS	REPORTER	TROUBLESHOOT
POWER	REPRESENTATIVE	TROUBLESHOOTING
POWERED	REQUIRED	
PRODUCT	RESERVOIR	UNIT
PRODUCTS	RESOLVED	UNKOWN
PROPERLY	RETURNED	USE
PROVIDED	REVEALED	VENTILATOR
PT	SENT	VNS
PUMP	SERVICE	VOLTAGE
RAY	SITE	WENT
READING	SORIN	WORKING

APPENDIX F: CLASSIFIER PARAMETERS

The classifier classes in the scikit-learn module (Pedregosa et al., 2011) are highly configurable. The following table describes our use of the parameters governing the behavior of the classifiers:

Classifier	**Parameter**	**Meaning**	**Value**
Stochastic Gradient Descent with Support Vector Machine	alpha	Constant regularization multiplier used to compute the learning rate.	0.0001
	class_weight	Weight of the classes. We assign equal weight (default of 1) to both classes.	1
	fit_intercept	Indicates if the intercept should be calculated and used by the model. When the value is False, the model does not calculate or use the y-intercept term. That assumes the data is centralized (i.e. $y = 0$). Value of True causes the model to calculate the intercept and use it in the model.	True
	l1_ratio	Elastic net regression parameter that blends the L1 and L2 norms for penalization. Value between 0 (L2) and 1 (L1).	0.15

Classifier	Parameter	Meaning	Value
	learning_rate	Learning rate for the stochastic gradient descent. Value of 'optimal' indicates a decaying scheme over training steps (large adjustments early, but gradually smaller as the model reaches convergence).	'optimal'
	loss	Loss function to be used in training. 'hinge' means a linear Support Vector Machine.	'hinge'
	max_iter	Maximum number of passes while the fitting the training data. Defaults to 5.	5
	n_jobs	Number of CPUs to use in the computation.	1
	penalty	Regularization term, that penalizes the model to reduce overfitting.	l2
	random_state	Seed value for the pseudo random number generator. A value of None indicates the system default (e.g. /dev/urandom)	None
	shuffle	Indicates if the raining data should be shuffled.	True

Classifier	Parameter	Meaning	Value
	tol	Tolerance for the stopping criteria during training. If the value is not None, the training stops when $current\ loss\ >\ previous\ loss - tolerance\ value$. If the value is None, training stops only when $current\ loss\ >\ previous\ loss$.	None
	warm_start	Indicates if the model should reuse the feature coefficients from the previous fit as initializing values. The value of False indicates the previously fitted coefficients are not used)	False
Logistic Regression	c	Inverse of regularization strength used to decrease the magnitudes of parameters and overfitting.	1.0
	class_weight	Weight of the classes. We assign equal weight (default of 1) to both classes.	None
	dual	Formulation (primal or dual) for the programming problem. Value of False indicates a primal formulation and is recommend for training use cases	False

Classifier	Parameter	Meaning	Value
		where the number of samples is greater than the number of features.	
	fit_intercept	Indicates if the intercept contestant should be used in the decision function.	True
	intercept_scaling	Intercept scaling multiplier for synthetic i.e. scaled) feature weights.	1
	multi_class	Parameter indicating the classification problem (i.e. multinomial or one-versus-rest). The value of 'ovr' indicates each class should be fitted as a binary classification problem.	'ovr'
	penalty	Norm used in penalization during training.	'l2'
	random_state	Seed value for the pseudo random number generator.	1
	solver	Optimization algorithm. The value of 'liblinear' tells the classifier to use the LIBLINEAR algoritm (Fan, Chang, Hsieh, Wang, & Lin, 2008).	'liblinear'
	tol	Tolerance for the stopping criteria during training. Training stops when	0.0001

Classifier	Parameter	Meaning	Value
		$current\ loss >$ $previous\ loss - tolerance\ value.$	
Multinomial Naïve Bayes	alpha	Additive smoothing parameter.	1.0
	class_prior	Fixed prior probabilities of the classes. The value of None indicates the prior probabilities are calculated according to the training data.	None
	fit_prior	Indicates whether the model should derive and use class prior probabilities. A value of True means class prior probabilities are used.	True

APPENDIX G: MOST INFORMATIVE FEATURES

Each of the three classifiers (Naïve Bayesian, Logistic Regression and Support Vector Machine) we selected for this research was trained on a term-frequency inverse-document-frequency (TF-IDF) matrix of features. The following table lists the 50 most informative positive features (i.e. with most significant weight) and 50 most informative negative features reported by each of the classifiers after the training.

Classifier: Multinomial Naïve Bayes

Negative Features	
Weight	**Feature**
-12.1	ACETABULAR
-12.1	ALVAL
-12.1	ANEURYSM
-12.1	ARTERIOTOMY
-12.1	BIOPROSTHETIC
-12.1	CALCIFIED
-12.1	CHROMIUM
-12.1	CIRCUMFLEX
-12.1	COINCIDENT
-12.1	COOK
-12.1	COUNSEL
-12.1	CYPHER
-12.1	DILATATION
-12.1	DISLOCATION
-12.1	DISLODGEMENT
-12.1	DISSECTION
-12.1	DOMESTICALLY
-12.1	DYSPAREUNIA
-12.1	ELUTING
-12.1	ENDOLEAK
-12.1	EXPRESS2
-12.1	EXTERNALIZED
-12.1	FEMUR

Positive Features	
Weight	**Feature**
-3.85	ISSUE
-3.88	ERROR
-4.18	EVENT
-4.4	DISPLAY
-4.44	ALARM
-4.44	DEVICE
-4.46	INDICATION
-4.57	METER
-4.68	KEYPAD
-4.73	BUTTON
-4.91	TESTED
-4.94	SOFTWARE
-4.98	REPORT
-4.98	USER
-4.99	SCREEN
-5.03	ANALYSIS
-5.07	BATTERY
-5.11	EVALUATION
-5.12	INFORMATION
-5.14	MESSAGE
-5.15	STATED
-5.16	DISPLAYED
-5.16	TIME

Negative Features	
Weight	**Feature**
-12.1	FILMS
-12.1	FRACTURES
-12.1	ILIAC
-12.1	INCONTINENCE
-12.1	INFLAMMATION
-12.1	INTERNATIONALLY
-12.1	IONS
-12.1	LAD
-12.1	LIBERTE
-12.1	LINER
-12.1	LITIGATION
-12.1	LV
-12.1	METALLOSIS
-12.1	MODERATELY
-12.1	NOVO
-12.1	OSTEOLYSIS
-12.1	PAPERS
-12.1	PFS
-12.1	PINNACLE
-12.1	POLY
-12.1	POLYETHYLENE
-12.1	PROGLIDE
-12.1	PROLAPSE
-12.1	PROMUS
-12.1	RA
-12.1	RCA
-12.1	RESTENOSIS

Positive Features	
Weight	**Feature**
-5.2	MEDICAL
-5.2	DELIVERY
-5.24	ASSOCIATED
-5.3	TEST
-5.31	HISTORY
-5.31	BOOT
-5.32	USA
-5.32	OCCURRED
-5.35	MOTOR
-5.35	BOARD
-5.4	BUTTONS
-5.49	CODE
-5.5	CONDITION
-5.53	PHONE
-5.53	INTERFACE
-5.54	MODULE
-5.57	REVIEW
-5.59	LOG
-5.6	ALARMED
-5.61	UNKNOWN
-5.62	TUBE
-5.65	WINDOW
-5.66	PLAN
-5.66	REPLACEMENT
-5.67	UNRESPONSIVE
-5.67	REVERT
-5.69	VERSION

Classifier: Logistic Regression

Negative Features	
Weight	**Feature**
-7.74	LEAD

Positive Features	
Weight	**Feature**
14.79	SOFTWARE

Negative Features	
Weight	**Feature**
-5.35	SUPPLEMENTAL
-5.18	PAIN
-5.11	INFORMATION
-5.07	LOT
-4.81	SEPARATE
-4.78	REVISION
-4.77	REVISED
-4.46	ALLEGATIONS
-4.42	BROKEN
-4.39	DEPUY
-4.34	FILED
-4.27	CATHETER
-3.94	PROCEDURE
-3.88	IMPLANTED
-3.84	STENT
-3.82	CONCLUSION
-3.82	LEAK
-3.8	CASING
-3.54	SUBMITTED
-3.48	IMPLANT
-3.4	BROKE

Positive Features	
Weight	**Feature**
14.51	ERROR
10.49	DISPLAY
9.48	ALARM
8.34	SCREEN
7.95	BOARD
7.52	HISTORY
6.98	BOOT
6.53	KEYPAD
6.49	BUTTON
5.67	ISSUE
5.6	COMPUTER
5.47	IMAGE
5.43	ALARMS
5.25	DISPLAYED
5.15	MEMORY
5.06	BUTTONS
4.96	ALARMED
4.96	TOUCHSCREEN
4.9	MESSAGE
4.86	MONITOR
4.76	PROGRAMMING

Negative Features	
Weight	**Feature**
-3.31	BRAKE
-3.26	POST
-3.21	LEAKING
-3.12	USED
-2.97	UPDATED
-2.96	CURRENTLY
-2.88	INFECTION
-2.86	LITIGATION
-2.84	LOOSE
-2.82	ZOLL
-2.81	MEDWATCH
-2.79	PENDING
-2.78	COMPLETE
-2.76	SURGERY
-2.75	RESULTS
-2.74	RECORDS
-2.62	INR
-2.59	DEVICES
-2.59	COMPLICATIONS
-2.58	SAMPLE

Positive Features	
Weight	**Feature**
4.6	LOCKED
4.51	METER
4.51	LOG
4.4	HARD
4.22	IMAGES
4.11	SHUT
4.08	TESTED
3.99	OCCURRED
3.99	FIRMWARE
3.96	OPERATING
3.87	PCB
3.86	PHONE
3.77	UNRESPONSIVE
3.74	INTERMITTENTLY
3.69	SETTINGS
3.62	USB
3.57	PLAN
3.56	ENGINEER
3.56	INCORRECT
3.56	USER

Negative Features	
Weight	**Feature**
-2.56	BATCH
-2.54	LEAKED
-2.53	COMPLETION
-2.45	HIP
-2.44	INDICATOR
-2.43	EVAL
-2.42	SUBJECT
-2.39	BOSTON

Positive Features	
Weight	**Feature**
3.55	CODE
3.52	STATED
3.51	RECEIVER
3.5	DELIVERY
3.45	REBOOTED
3.44	MINUTES
3.44	HANDHELD
3.41	RESET

Classifier: Support Vector Machine with Stochastic Gradient Descent

Negative Features	
Weight	**Feature**
-2.44	LEAD
-1.78	LOT
-1.78	INFORMATION
-1.73	SEPARATE
-1.57	SUPPLEMENTAL
-1.51	PAIN
-1.44	CASING
-1.35	PROCEDURE

Positive Features	
Weight	**Feature**
4.73	ERROR
4.63	SOFTWARE
3.65	DISPLAY
3.03	ALARM
3.02	SCREEN
2.75	ISSUE
2.66	BOARD
2.27	HISTORY

Negative Features	
Weight	**Feature**
-1.34	LEAK
-1.32	BROKEN
-1.32	IMPLANT
-1.27	SUBJECT
-1.24	FILED
-1.23	CATHETER
-1.2	REVISION
-1.15	SURGERY
-1.14	CONCLUSION
-1.13	USED
-1.12	BRAKE
-1.11	INR
-1.11	DEPUY
-1.11	DEALER
-1.06	LEAKING
-1.06	BROKE
-1.03	PENDING
-1.03	RECORDS
-1.01	STENT
-1	SUBMITTED
-0.98	IMPLANTED

Positive Features	
Weight	**Feature**
2.23	BOOT
2.21	BUTTON
2.14	DISPLAYED
2.11	METER
2.08	IMAGE
2.01	MONITOR
1.99	KEYPAD
1.96	COMPUTER
1.91	ALARMS
1.9	TESTED
1.88	TOUCHSCREEN
1.85	ALARMED
1.81	SHUT
1.78	MESSAGE
1.73	ENGINEER
1.72	PROGRAMMING
1.72	STATED
1.7	OCCURRED
1.66	LOCKED
1.65	IMAGES
1.65	BUTTONS

Negative Features	
Weight	**Feature**
-0.97	POST
-0.97	REVISED
-0.93	DEVICE
-0.92	COMPLICATIONS
-0.92	INFECTION
-0.92	UPDATED
-0.91	CURRENTLY
-0.9	BATCH
-0.9	ERRONEOUS
-0.9	SAMPLE
-0.89	EVALUATE
-0.89	RESULTS
-0.85	PASS
-0.85	LEGAL
-0.85	INRATIO
-0.85	BOSTON
-0.84	SCIENTIFIC
-0.84	DEVICES
-0.84	EXPLANTED
-0.82	LOOSE
-0.81	HOLDING

Positive Features	
Weight	**Feature**
1.62	OPERATING
1.54	USER
1.53	CABLE
1.51	INTERMITTENTLY
1.5	FIRMWARE
1.48	UNRESPONSIVE
1.48	PHONE
1.47	LOG
1.47	MODULE
1.47	FLUORO
1.47	MEMORY
1.46	BATTERY
1.45	PROGRAMMER
1.4	SWITCH
1.4	BATTERIES
1.4	CODE
1.39	REBOOTED
1.38	MONITORS
1.38	CALIBRATION
1.37	CINE
1.36	PCB

APPENDIX H: SAMPLE OF EVENTS WITH COMPUTING TECHNOLOGY CAUSES

To assess the effectiveness of the problem code assignment published in the MAUDE datasets, we first obtained a sample of events with strong computer technology causality. The method for this sampling is discussed in Section 4.9.1. The table below lists the identifiers for a small sample of reports of events with computer technology causality.

MDR_REPORT_KEY	MDR_TEXT_KEY
879117	641235
997881	8091319
1214830	957549
1220838	957998
1455879	19206289
1571353	8425432
1711538	1536408
1819444	16412636
1931462	16705771
2066189	9115241
2216497	9429496
2275187	18202714
2302306	19652874

MDR_REPORT_KEY	MDR_TEXT_KEY
2314896	21975912
2352682	18050812
2359547	9898715
2359639	2379635
2407240	2539386
2449480	2482860
2506612	21921360
2604215	2745043
2798018	10266662
2858634	2953139
2886330	10365869
2937454	3235414
2994139	3171553

MDR_REPORT_KEY	MDR_TEXT_KEY
3022001	3405849
3031268	10592287
3256955	15282948
3320545	27443146
3328297	11169817
3341812	17618620
3341882	15016980
3448598	21490240
3448716	21966694
3448744	3737881
3448745	20637474
3448804	19559197
3466409	11410578
3742972	12163209
3795936	15550554
3823329	11873232
3995715	12441074
3996879	4781083
4250870	16439876
4291942	13245332
4317222	22038450

MDR_REPORT_KEY	MDR_TEXT_KEY
4489059	49241068
4503014	19089124
4670977	5692728
4726159	13150947
4726879	5745140
4727612	13492962
4808501	61000788
4853057	23844936
4860667	36609534
4862826	32382997
4885503	16904832
4891913	24088040
4973167	38098239
4992244	22710293
4998435	28944472
5016637	24576644
5040100	24544422
5081947	34207416
5087987	26177195
5130337	27923740
5144030	31757276

MDR_REPORT_KEY	MDR_TEXT_KEY
5173671	29209706
5234357	36727604
5265912	63969925
5290339	33865986
5296801	38228660
5304430	34325554
5315750	34443726
5367733	39129807
5383602	39562903
5394642	41742714
5481181	42058548
5493009	43296050
5518232	42860044
5563324	53000465
5601982	50404243
5611836	43780253
5612260	48201420

MDR_REPORT_KEY	MDR_TEXT_KEY
5621943	44567390
5659693	46385684
5667071	51085826
5690889	48656426
5703503	47201427
5742098	53711328
5820852	53467140
5886040	53062694
5895179	55047550
5940750	60681135
5954311	54941314
5980174	56056538
5992407	58370798
6038935	57807126
6066495	58742033
6089282	59737217
6132784	61296450

APPENDIX I: COMPUTING ENVIRONMENT

This research involved extensive utilization of various software applications and libraries across multiple computing environments. Tables below list the key applications and libraries we used, and their versions:

Machine Learning Environment:

Software Component	**Version**	**Purpose**
Ubuntu	16.04.2 LTS	Operating System
Python	3.5.2	Runtime for machine learning experiments
Scikit-Learn	0.19.1	Python module for machine learning
SciPy	1.0.0	Python module for math, science and engineering
NLTK	3.2.5	Python module for natural language processing
NumPy	1.13.3	Python module for multi-dimensional matrix and array manipulation

Data Analysis Environment:

Software Component	Version	Purpose
Microsoft Windows 10	64-bit Build 16299.125	Operating system
Microsoft Excel	2016 Build 8827.2148	Data analysis, graphing and plotting
Microsoft SQL Server	2016 Developer Edition (64-bit) 13.0.4206.0	Large scale data storage and query

Software Development Environment:

Software Component	Version	Purpose
Microsoft Windows 10	64-bit Build 16299.125	Operating system
Visual Studio	2017 (v15.2) Community Edition	Python source code editor
PyCharm	2017.2 Community Edition	Python source code editor
Microsoft SQL Server Management Studio	13.0.16106.4	T-SQL source code editor
Git	2.11.0.windows.1	Source control

Documentation Environment:

Software Component	Version	Purpose
Microsoft Windows 10	64-bit Build 16299.125	Operating system
Microsoft Word	2016 Build 8827.2148	Document editor
EndNote	X8.1 Build 11010	Bibliography manager
Notepad++	7.5.4	Text editor

Repositories:

We deposited all of the source code we created, and the results data we produced as a part of this research into a set of cloud-based repositories. While the permanent availability of these external repositories is not guaranteed, they may be accessed from the following uniform resource locators while available:

Source Code:

https://github.com/dkhanal/maude_experiments/tree/dissertation-final

Results Data (Raw and Synthesized):

https://maude.research.dkhanal.com/final-results/index.html