Using Machine Learning Predictive Models to Aid Manufacturers Identify Early Warning Signs of Ventilator Defects

by Valerie I. Santiago-Pérez

B.S. in Industrial and Manufacturing Systems Engineering, May 2015, Iowa State University, Iowa
M.B.A in Management and Leadership, May 2017, Texas A&M University-Central Texas, Texas

A Praxis submitted to

The Faculty of
The School of Engineering and Applied Science
of The George Washington University
in partial fulfillment of the requirements
for the degree of Doctor of Engineering

May 15, 2022

Praxis directed by

Oliver Bojanić
Professorial Lecturer of Engineering Management and Systems Engineering

The School of Engineering and Applied Science of The George Washington University certifies that Valerie I. Santiago-Pérez has passed the Final Examination for the degree of Doctor of Engineering as of March 31, 2022. This is the final and approved form of the Praxis.

Using Machine Learning Predictive Models to Aid Manufacturers Identify Early Warning Signs of Ventilator Defects

Valerie I. Santiago-Pérez

Praxis Research Committee:

Oliver Bojanić, Professorial Lecturer of Engineering Management and Systems Engineering, Praxis Director

Amir Etemadi, Associate Professor of Science and Engineering, Committee Member

J.P. Blackford, Professorial Lecturer of Engineering Management and Systems Engineering, Committee Member

Dedication

I would like to dedicate this work to my family. Without your continuous support, encouragement or words of advice this would have not been possible. To my husband—thank you, thank you, thank you for everything. I love you so much more for sacrificing all those long nights in helping me complete this process. To my 5 babies, Julián, Valentina, Marley, Abbie, and Lulu—thank you for being patient with me and providing the motivation I needed at all times. To Papi, Mami, Ricky and Diego—thank you for pushing me towards the finish line. To Antonio and Jackie—thank you for supporting my goals and taking part in all my achievements. And to everyone else who was there—thank you for helping in some form or another, as this has been a long journey and it was definitely made possible by all those helping hands along the way.

Acknowledgements

I would like to acknowledge the tremendous amount of patience, guidance and help that was given to me by my advisor, Dr. Oliver Bojanić. Thank you for your knowledgeable advice and expertise as I completed this praxis. In addition, I would also like to acknowledge the Subject Matter Experts (SMEs) team that contributed their time to this project. Your contributions are invaluable and worthy of praise. Thank you.

Abstract of Praxis

Using Machine Learning Predictive Models to Aid Manufacturers Identify Early Warning Signs of Ventilator Defects

The United States (U.S.) Food and Drug Administration (FDA) maintains a Manufacturer and User Facility Device Experience (MAUDE) database containing end-user experience reports, which detail adverse events encountered with medical devices. However, there is currently no process for efficiently mining through the millions of reports available. Furthermore, the current Coronavirus Disease 2019 pandemic has exacerbated the need for an innovative technique to preemptively detect device malfunctions for essential medical devices, such as ventilators. This need was evidenced by a recent mass recall of faulty ventilators, one of the most crucial devices through the pandemic, costing the manufacturer millions of dollars and debilitating an already-frail supply chain.

This praxis explores a combined machine learning approach utilizing Latent Dirichlet Allocation (LDA) topic modeling and predictive models for text classification—including Logistic Regression (LR), Gaussian Naïve Bayes (NB), Random Forest (RF), Adaboost (AB) and Support Vector Machines (SVM) with Stochastic Gradient Descent training (SGD)—to uncover early warning signs of ventilator defects and to predict potential recalls and their associated recall classifications. The proposed methodology presented in this praxis introduces a novel approach for ventilator defect detection and prediction of ventilator recalls. Using LDA topic modeling, early warning signs of ventilator defects are extracted from relevant adverse event narrative reports included in the MAUDE database and then utilized as

feature vectors for recall status determination through predictive modeling. Finally, a text classification model is employed to determine the applicable recall classification level of ventilators that have been predicted to be recalled.

The results from this research indicate that ventilator manufacturers and design engineers can effectively uncover early warning signs of ventilator defects and proactively develop recall mitigation strategies using the proposed methodology.

Table of Contents

List of Figures

List of Tables

List of Symbols

α per-record topic distributions in LDA

β per-topic word distribution in LDA

θ topic distribution for record m in LDA

φ word distribution for topic k in LDA

z topic for the n-th word in record m in LDA

w specific word in LDA

D_n set of n records in Text Classifiers

C_n set of labels that differentiate each class of the record D in Text Classifiers

i total number of labels in Text Classifiers

f classifier model in Text Classifiers

T training set in Gini Index

x topic distributions in Gaussian Naïve Bayes

i single record in Gaussian Naïve Bayes

μ mean

σ^2 variance

N number of features in Logistic Regression

x_i input vector in Logistic Regression

w_i weight for input vector in Logistic Regression

List of Acronyms

U.S.	United States
FDA	Food & Drug Administration
MAUDE	Manufacturer and User Facility Device Experience
EUA	Emergency Use Authorizations
PMA	Premarket Approval
MDTD	Medical Device Technologies' Design and Development process
MDTAU	Medical Device Technology Amelioration & Upgrading process
QMS	Quality Management Systems
LDA	Latent Dirichlet Allocation
SME	Subject Matter Expert
NLP	Natural Language Processing
tf-idf	Term Frequency Inverse Document Frequency
POS	Part-of-Speech Tagging
PCA	Principal Component Analysis
SGD	Stochastic Gradient Descent
RF	Random Forest
SVM	Support Vector Machine
NB	Gaussian Naïve Bayes
AB	AdaBoost
LR	Logistic Regression

Chapter 1—Introduction

1.1 Background

Post-market surveillance data for medical devices are extremely valuable for manufacturers and regulatory agencies. While pre-market controls help anticipate potential device issues, rigorous post-market surveillance is required to increase patient safety and mitigate the risks associated with non-compliance for manufacturers, such as product recalls and financial repercussions (Blake, 2013). Thorough analysis of information provided by end users on their experiences with medical devices allows manufacturers to implement design changes and prevent major product recalls (Vockley, 2015). To this end, the Food and Drug Administration (FDA) website houses two major databases containing information about adverse health events and recall notices for medical devices. One of these is the Manufacturer and User Facility Device Experience (MAUDE) database, which provides adverse event reports submitted by patients, healthcare professionals, and customers and allows for the monitoring and research of issues related to medical devices (Ensign & Cohen, 2017). The second is the Medical Device Recall database, which serves to notify end users about devices that have been recalled and need to be modified or removed entirely from the market.

Along with these notices, the FDA assigns recall level classifications to faulty medical devices after a recall event to warn users of their level of risk if used. These classifications (Class I, II, or III) signify the degree of health hazard a recalled device can present to the end user. As described by the FDA, Class I recalls indicate that there is reasonable chance that a medical device will lead to serious health consequences, or even

death. Class II recalls indicate that the medical device could cause temporary health consequences, but the potential for more serious health issues is low. Class III recalls are reserved for medical devices that are highly unlikely to lead to adverse health issues (Center for Devices and Radiological Health, 2020). Even with adverse event report data readily available, however, medical device defects still go undetected and potentially require recalls, leading to catastrophic financial impacts to manufacturers and injury or even death for the products' end users (Livingston & Mattingly 2021).

During the current Coronavirus Disease 2019 pandemic, ventilator manufacturers have faced increased pressure from regulatory agencies and the general public, as demand for their life-sustaining products has skyrocketed (Billingham, Widrick, Edwards, & Klaus, 2020). In the state of New York, for example, shortly after the pandemic began in March 2020, the governor ordered 17,000 ventilator units to satisfy the needs of hospitals, but was only allocated 2,500 in the two weeks following his request due to worldwide shortages (Mehrotra, Rahimian, Barah, Luo, & Schantz, 2020). These scarcities led to the issuance of 86 ventilator Emergency Use Authorizations (EUAs), in an effort to meet the needs of patients with severe respiratory complications (Center for Devices and Radiological Health, 2020). As described by the FDA, EUAs authorize the unapproved use of medical devices to treat life-threatening diseases or conditions (Office of the Commissioner, 2021). These EUAs also allow manufacturers to modify already-existing medical devices by replacing parts or materials during production to accelerate the assembly process, further increasing the inherent risk of use and the potential for a recall (Badnjević, Pokvić, Džemić, & Bečić, 2020).

Major ventilator manufacturer Philips Respironics experienced this risk first hand, as a massive recall was issued for a variety of its devices in June 2021, leading to a flood of lawsuits and diminished public trust (Center for Devices and Radiological Health, 2021). In a statement to inform end users of the manufacturer's response efforts, medical experts explained that the recall event that affected 3–4 million devices worldwide was not appropriately addressed by Philips Respironics, leaving millions of patients at risk (Owens, Wilson, Gurubhagavatula, and Mehra, 2021). As a result of this recall, the company reported in its second-quarter report on July 26th, 2021 a major drop in income (down to EUR 65 million) from continuing operations in the respiratory device sector compared to the income level for this division in the second quarter of 2020 (EUR 195 million; Philips, 2021).

As discussed by Fu et al. (2017), consistent monitoring of feedback provided by end users through the MAUDE database allows manufacturers and healthcare professionals to anticipate potential recalls. Thus, it stands to reason that Philips could have mitigated the negative impact of their ventilator recall through more effective post-market surveillance efforts.

1.2 Research Motivation

Timely analysis of user feedback data grants manufacturers early insights into product defects, thereby improving their product quality management processes (Zheng, He, & He, 2020). With the amount of adverse event narrative reports submitted to the MAUDE database having grown exponentially in recent years—from 635,654 in 2010 to 2,965,324 in 2020—it is now even more difficult for manufacturers to mine through heaps of data (Center for Devices and Radiological Health, 2020). The staggering amount

of reports that continue to be submitted indicates a need for advanced analysis methods that are able to predict medical device recalls on a real-time basis (Mukherjee & Sinha, 2018). In addition, as a consequence of the Coronavirus Disease 2019 pandemic, manufacturers have faced unprecedented demand spikes that have hindered their ability to accurately predict production needs (Billingham, Widrick, Edwards, & Klaus, 2020). These demand surges have led manufacturers to accelerate their production times, which sacrifices quality and risks potential device recalls (Livingston & Mattingly, 2021).

Prior studies have demonstrated how improved digital technologies have paved the way for user feedback data to be used in detecting early warning signs that help predict product recalls (Bhat & Culotta, 2017; Fong, Sarkani, & Fossaceca, 2021). For example, within the field of children's products, Bhat and Culotta (2017) presented successful text classification models that are able to identify warning signs of product defects prior to a recall being issued. In the furniture manufacturing field, Fong, Sarkani, and Fossaceca (2021) introduced an approach combining recurrent neural network models and topic modeling to automatically detect product defects from online customer reviews in order to prevent recalls. As no other predictive models have been found in the literature that specifically address medical device recalls, this praxis builds upon the frameworks provided by prior studies to propose a text mining approach for the prediction of ventilator recalls and their associated classifications by identifying early warning signs of defects from adverse event narrative data.

The methodology presented in this study contains a combination of Latent Dirichlet Allocation (LDA) topic modeling and machine learning predictive models to identify early warning signs of ventilator defects that may lead to recalls. Predictive

modeling serves as the primary method of determining whether ventilators will be recalled as well as their associated classification level once recalled.

1.3 Problem Statement

It is difficult to identify early warning signs of ventilator defects from United States Food and Drug Administration data (Liebel et al., 2020), resulting in costly product recalls (Janetos, et al., 2017).

1.4 Thesis Statement

Predictive models utilizing topic modeling and machine learning will help ventilator manufacturers mitigate recalls by identifying early warning signs of defects.

1.5 Research Objectives

The primary focus of this study is to determine whether the adverse event narrative descriptions contained in FDA data provide early warning signs of defects that may lead to medical device recalls. This possibility is investigated by introducing a combination of LDA topic modeling and predictive models to target a series of research objectives. The initial objective is to identify early warning signs of ventilator defects by applying a topic modeling approach to the adverse event narrative data. From the trained LDA model, the second objective is to develop a classification model using the identified warning signs as feature vectors to predict if a ventilator will be recalled. The final objective is to develop a predictive model using adverse event raw narrative data to accurately forecast the recall classification level of ventilators that have been predicted to be recalled.

1.6 Research Questions and Hypotheses

RQ1: Can LDA topic modeling identify early warning signs of ventilator defects using adverse event narrative data?

RQ2: Can the identified early warning signs from the trained LDA model be used as feature vectors in classification models to predict the recall status of a ventilator?

RQ3: Can machine learning utilize adverse event raw narrative data to predict the recall classification levels of ventilator devices?

H1: Topic modeling will be able to identify 70% of early warning signs for ventilator defects using adverse event narrative data.

H2: The identified early warning signs can be used as feature vectors in classification models to predict potential recalls with 75% accuracy.

H3: Machine learning can predict the associated recall classification level of defective ventilators with 85% accuracy.

1.7 Scope of Research

The data utilized for this research comprise actual feedback provided by end users on separate ventilator types, with each type containing various ventilator models. The ventilator types included are a mixture of life-sustaining ventilators, such as continuous facility-use/home-use ventilators, and non-life-sustaining ventilators, such as non-continuous/high-frequency ventilators, as these devices have led to recalls of various classes. The final dataset used as input for this praxis contains 3,992 adverse event

reports that were randomly selected from the available ventilator user feedback data for the specified device types and recall classification levels.

The final product of this research consists of a list of early warning signs for ventilator defects obtained from an LDA topic model and a series of predictive models trained to predict recall status and associated recall classification level. Latent Dirichlet Allocation topic modeling initiates the process by extracting a series of topics from narrative reports submitted to the FDA by end users regarding adverse events encountered with ventilators. After the topics and their associated words are validated as early warning signs of ventilator defects by a panel of subject matter experts (SMEs), they are used to predict the recall status of ventilators via machine learning algorithms. Finally, for ventilators that are predicted to be recalled, classification into the three separate recall classes (Class I, II, or III) is predicted, helping determine the risk level associated with continued use of the product after it has been recalled. The predictive models employed for these applications include Logistic Regression, Gaussian Naïve Bayes, Random Forest, Adaboost, and Support Vector Machines with Stochastic Gradient Descent training. The machine learning model results are compared within the individual applications and the highest-performing model is chosen for each phase using traditional model performance metrics, such as model accuracy and the macro F-1 score. All models are trained, tested, and validated using historical data from 2012 to 2019 that have been made publicly available by the FDA.

1.8 Research Limitations

The data available for this research are limited to the feedback entries submitted by end users in the FDA database. While there may be additional user feedback platforms

or forums, the models built in this research are designed to only interpret user experience reports included within the FDA database.

In addition, as recalled medical devices are classified using a three-level scale by the FDA to warn users of their risk of continued use, all recall classification levels are represented in the final dataset used. However, this inclusion introduces a class imbalance into the final classification phase of the praxis, which is addressed by stratified sampling prior to model training.

1.9 Organization of Praxis

This praxis is organized into five chapters. Chapter 1 introduces the research and provides a thorough background of the problem at hand. This chapter subsequently presents the underlying motivation for the research, the research questions and hypotheses, and finally the scope and limitations of the study. Chapter 2 provides detailed descriptions of related work on product defect and recall prevention, including studies specifically pertaining to the medical device field. This chapter explores current approaches to text mining and predictive modeling of user feedback data and reviews how manufacturers have benefitted from their application. Chapter 2 also introduces and explains the application of machine learning predictive models and topic modeling. Chapter 3 presents the research methodologies utilized in this praxis to address the research questions. Chapter 4 then presents the research results. Finally, Chapter 5 draws final conclusions and closes the praxis by discussing future research directions.

Chapter 2—Literature Review

2.1 Introduction

This chapter begins by introducing the evolution of the medical device industry and its regulatory environment in an effort to explain the changing nature of the field and the hurdles manufacturers encounter when developing new products. The chapter then explains medical device post-market surveillance and manufacturers' use of the data. The chapter presents the most widely used standard for quality management of medical devices and discusses the implications of device defects and recalls. Product recall strategies and the influences they have on manufacturers and stockholders of recalled products are explained. Studies relating to medical device design and defects are discussed, and special consideration is given to existing literature related to ventilator devices.

The concepts of Natural Language Processing (NLP) and text mining are then introduced as a precursor to the in-depth discussion of topic modeling and text classifier models. The concept of Latent Dirichlet Allocation (LDA) and its usability within various industries is introduced as a viable topic modeling technique to identify latent topics in textual datasets and detect product defects. Commonly used text classification algorithms are presented, with a brief segment on model performance enhancing techniques. Model evaluation metrics are also discussed, along with their corresponding mathematical computations. Finally, the chapter concludes by providing a summary of findings and the research contributions of this praxis.

2.2 Medical Device Industry

Over time, the medical device industry has been subject to countless changes driven by clinical needs, regulatory requirements, and technological advancements. The rapidly evolving nature of medical devices has led the United States Food & Drug Administration (FDA) to develop 6,500 medical device product categories (FDA, 2020). Medical device manufacturers undergo intense scrutiny from the FDA before and after a medical device is cleared to enter the market. Although the FDA's involvement in the medical device oversight process began in 1906, the current process utilized to monitor and regulate medical devices was more recently developed (Center for Devices and Radiological Health, n.d.). With the 1976 Medical Device Amendments to the Federal Food, Drug, and Cosmetic Act, Congress attempted to develop a regulatory model that would control medical device development without stifling innovation (Merrill, 1994). As part of its contributions, the Medical Device Amendments highlighted four major regulatory decisions: (1) the implementation of a risk-based, three-level medical device classification system; (2) three-tier regulatory controls, where each classification level is subject to specific controls; (3) "comparable regulations of old and new devices," according to which new medical devices that are considerably similar to existing devices cannot be denied immediate entry into the market; and (4) the establishment of regulatory frameworks to differentiate new medical devices from those already in the market: premarket approval (PMA) and premarket notification (510(k); Merrill, 1994). As this praxis is centered around defect detection for ventilators, **Table 2-1** defines the risk levels associated with the three tiers for FDA medical device classification using ventilators and their associated components as examples.

Medical Device FDA Classification	Risk Level	Examples
Class I	Low	Tubing and support sets for ventilators
Class II	Moderate	Continuous facility-use ventilators
Class III	High	High-frequency ventilators

Table 2-1. FDA Medical Device Classifications (sourced from Center for Devices and Radiological Health, n.d.)

2.2.1 Medical Device Design and Development Process

At the core of the medical device design process is human factors engineering. The methods utilized in human factors engineering consider all important aspects of product development, such as the environment in which the product will be used and the specific needs of its end users (Money et al., 2011). By studying the needs of end users of medical devices and the specific contexts in which the devices will be used, manufacturers are better prepared to design devices that fit the end user, instead of the other way around (Scanlon, Karsh, & Densmore, 2006). Furthermore, end-user involvement in the medical device design phase has been shown to limit redesign and reduce development costs (Bühler, Hoelper, Hoyer, & Humann, 1995). From the medical device design and development process, adapted from Motyl and Flippi (2014) and illustrated in **Figure 2-1**, it is clear that the design phase for a new medical device is extensive and continues through the development stage.

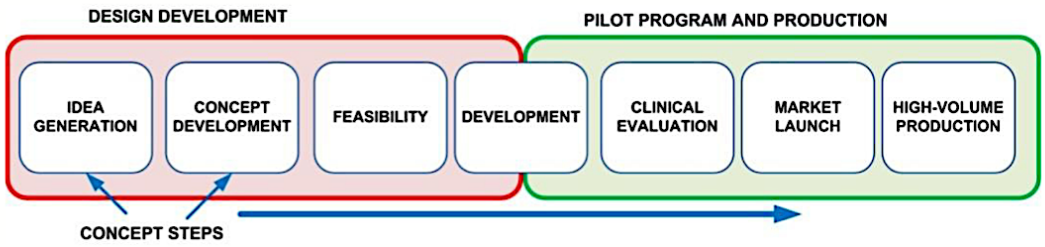

Figure 2-1. Medical Device Design and Development process (Motyl & Flippi, 2014

In 2016, Hani and de Marcellis-Warin conducted a study to determine at which stage of the medical device technologies' design and development (MDTD) process end-user involvement should be most prevalent to showcase its importance in the product life cycle. Their initial findings indicated that most end users (i.e., doctors and nurses) were interested in actively participating in all phases of the design and development process—including concept stages, device testing phases, training of medical professionals, and sharing their own experiences with initial prototypes to facilitate market acceptance (Hani & de Marcellis-Warin, 2016). Most end-user participants believed that their consistent involvement would reduce the need for product reengineering and subsequently reduce manufacturers' expenditures. Their final recommendation for the revised MDTD process, shown in **Figure 2-2**, depicts the shift to a medical device technology amelioration and upgrading (MDTAU) process, whereby end user and market feedback are continuously incorporated into the medical device design process to improve product performance (Hani & de Marcellis-Warin, 2016).

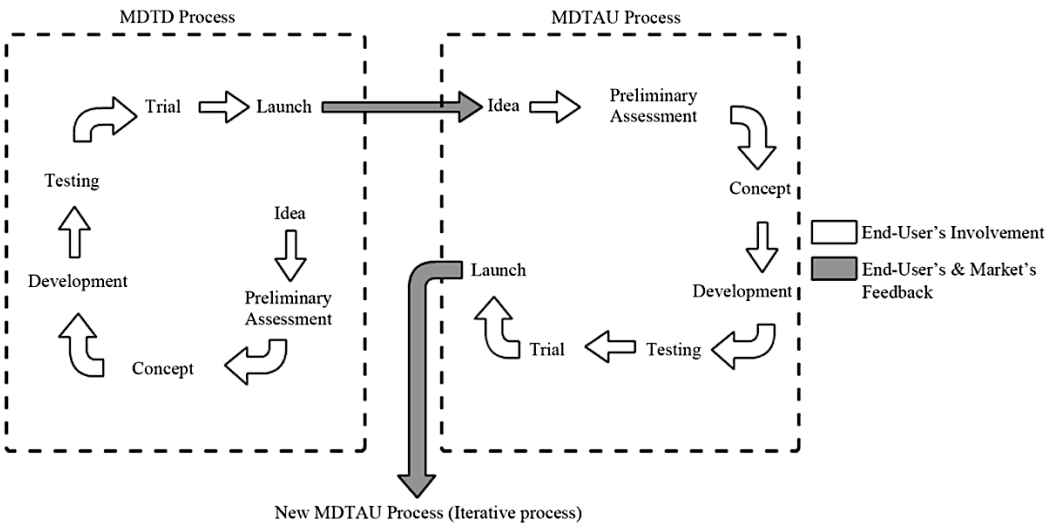

Figure 2-2. Medical Device Technology Amelioration & Upgrading Process (Hani and de Marcellis-Warin, 2016)

A study conducted by Markiewicz et al. (2017) explored the usability of this model's user-centered approach for medical device design as the ideal method to be adopted by medical device manufacturers. The researchers conducted 37 semi-structured interviews with participants from 36 individual medical device manufacturers to determine which methodologies were being used to develop new medical devices. Of the 37 participants, 25 reported successfully incorporating end-user feedback in the device idea-generation stage, similar to the MDTAU's recommendations (Markiewicz et al., 2017).

2.2.2 Ventilator Device Design

The design process for ventilators dates back to the mid-1500s, when manual ventilation methods such as mouth-to-mouth and mouth-to-nose were first documented as a means to supply oxygenation to patients undergoing respiratory failure (Khoury et al.,

2014). It was not until the 20th century, however, with the introduction of the iron lung and the bag valve mask, that artificial ventilatory support was successfully utilized to address the issue of insufficient pressure generated by mouth-to-mouth ventilation (Barbarash, Smith, Godwin, & Sahn, 1990; Khoury et al., 2014). As noted by Chatburn and Mireles-Cabodevila (2013), in the late 1970s, ventilator device design consisted of individual components that could be easily disassembled and reassembled in a short amount of time. Today, ventilators are considered complex medical devices containing many components managed by high-level software.

The complexity behind the modern processes dictating ventilator device design has led to a severe deficit in the supply of these devices (Madekurozwa et al., 2021). The lack of ventilators for life-sustaining efforts has been historically exacerbated by pandemic events, such as the polio pandemic in the 1950s, the Ebola pandemic from 2014 to 2016, and the recent Coronavirus Disease 2019 pandemic (West, 2005; Suffredini, 2015; Madekurozwa, M. et al., 2021). Motivated by the need imposed by these shortages, Madekurozwa et al. (2021) proposed an innovative approach to ventilator device design that is simpler, involves fewer components, and is easier to operate by clinicians. Their prototype demonstrated an ability to meet the performance requirements of commonly used ventilators for a variety of treatment applications.

2.2.3 Medical Device Post-market Surveillance

Following the design and development process, if medical devices are cleared to enter the market, their performance is closely monitored through post-market surveillance. Per Section 522 of the Federal Food, Drug, and Cosmetic Act, the FDA is granted the ability to require manufacturers to conduct post-market surveillance of high-

risk devices at any point during the product's life cycle. In an effort to enhance post-market surveillance activities and increase end users' safety, the FDA firmly supports the use of big data analytics to maximize the potential of real-time data analysis (Vockley, 2015).

In a 2010 post-market surveillance study conducted by Resnic et al., clinical health data and patient outcome information for adults who underwent coronary interventions at Massachusetts inpatient facilities during a 4-year span was used to develop an automated system that would detect indicators of cardiovascular medical device defects. Through the use of statistical analyses and logistic regression models, the researchers found that low-frequency safety signals were detected in two of the seven devices studied—indicating that continued use of those devices led to an increased risk of major life-threatening vascular complications (Resnic et al., 2010). The methodologies explored by this study successfully demonstrated that automated surveillance systems can be applied to medical data registries to uncover safety signals for defective products. However, the study did not indicate how manufacturers could utilize these findings to mitigate potential faulty device recalls as part of their post-market surveillance efforts.

Callahan et al. (2019) similarly demonstrated that the use of deep learning methods applied to electronic health records could identify over six times as many hip implant-related complications when compared to the more traditional supervised learning models. Their model was able to extract device details and attributable complications with up to 96.3% precision, 98.5% recall, and an F-1 score of 97.4% *(see **Section 2.4.3** for an explanation of the model performance metrics)*, but did not indicate how manufacturers could utilize their model to prevent recalls (Callahan et al., 2019).

2.3 Medical Device Quality Management

The government website SelectUSA (https://www.selectusa.gov/medical-technology-industry-united-states) reported that the U.S. medical device industry is expected to grow to $208 billion in 2023 from $156 billion in 2017. Because of the demanding nature of the medical device industry, manufacturers are under intense pressure to design safe and reliable devices faster than ever. In 1996, the FDA published "Medical Devices; Current Good Manufacturing Practice (cGMP) Final Rule; Quality System Regulation," detailing the requirements and regulations that device manufacturers must follow to ensure that their products are compliant. While it is crucial for manufacturers to abide by all the regulations detailed in the cGMP Final Rule to remain compliant, their productivity levels must also remain unaffected.

Within the medical device industry, "ISO 13485 Standard" is widely used as the guiding principle of quality management systems (QMS; Hegde & Konakanchi, 2011). The requirements of this QMS are presented in the most current version, "ISO 13485:2016 Standard." With this standard, specific roles are designated within all levels of a medical device manufacturing organization to ensure that all groups are in compliance and the final product is produced to satisfaction (Abuhav, 2018).

2.3.1 Medical Device Defects

Despite the rigor of FDA regulations and the thoroughness of manufacturers' quality assurance efforts, medical devices are subject to failure, a costly reality that risks the lives of end users and jeopardizes manufacturers' images. Medical device failures can be categorized as manufacturing or functional defects, packaging errors, and/or software glitches (Thirumalai & Sinha, 2011). These device malfunctions vary in severity, from

mildly inconvenient to potentially life-threatening. While medical device manufacturers are expected to devote considerable resources to developing reliable products, the variety of defects that continually arise indicate that they may not be allocating enough (Thirumalai & Sinha, 2011). This failure is evidenced by the amount of adverse event narrative reports submitted to the FDA regarding malfunctioning devices, which has nearly doubled from 1.17 million reports received in 2012 to 2.81 million reports received in 2019 (Center for Devices and Radiological Health, n.d.).

2.3.2 Ventilator Defects

In the time periods studied by this praxis, the amount of adverse event reports received by the FDA for a single ventilator type, continuous facility-use ventilators, increased from 6,088 in 2012 to 9,142 in 2019 (Center for Devices and Radiological Health, n.d.). These defects could be attributed to a myriad of factors, including software issues, battery issues, power supply malfunctions, or incorrect product assembly (Krishna Kumar, Ravi, Dinesh, & Nanda, 2011). Once these reports are submitted, ventilator manufacturers are responsible for mining the MAUDE database to determine the root cause of their products' failure. To assist manufacturers in this process, this praxis aims to provide an additional resource for the automatic extraction of product defects from adverse event narrative reports using a subset of ventilator-related event reports submitted to the FDA.

While no previous studies have focused on text mining of the MAUDE database for early warning sign detection for ventilator defects, several studies have mined hospital-specific repository data to determine possible causes of failures reported for a

series of home ventilator devices (Srinivasan et al., 1998; Farre et al., 2005; Clemente, Faiella, Rutoli, Bifulco, Romano, & Cesarelli, 2019).

2.3.3 Medical Device Recalls

When medical devices fail, manufacturers may be forced to recall their product, depending on the severity of the malfunction. Medical device recalls can occur in two ways—as voluntary actions taken by manufacturers to correct devices that are noncompliant with FDA regulations or as mandated actions by the FDA if the manufacturer fails to recall voluntarily (Sarkissian, 2018). The former is more common; FDA-mandated recalls are reserved for urgent matters and are immediately directed to the manufacturer (Center for Devices and Radiological Health, n.d.). Regardless of how they are initiated, recalls are classified into one of three categories according to the risk level of the product's continued use after recall. As defined by the FDA, Class I recalls are the most severe, as this classification pertains to situations in which the "violative product will cause serious adverse health consequences or death." Class II recalled devices could lead to temporary or reversible health conditions, and Class III recalled devices are not likely to lead to adverse health conditions (Center for Devices and Radiological Health, n.d.).

The FDA maintains a database of medical devices that have been recalled through the years. Recall trends for recent years indicate that there has been a considerable increase in quality issues encountered by end users. In fact, the FDA reported an increase to approximately 596 million recalled medical devices between 2018 and the first half of 2019 from the 410 million medical devices that were recalled between 2016 and 2017 (Center for Devices and Radiological Health, n.d.).

While the FDA assigns classifications to recalled medical devices according to the risk of continued use, the implications for immediate market withdrawal for high-risk recalls can also be harmful to patients. For cases in which recall events affect a device that is widely used in a life-sustaining manner or in cases in which only one manufacturer produces the device, the risk of market removal may affect patient safety even more than its continued use (Caceres, 2014).

2.3.4 Medical Device Recall Strategies

Regardless of whether or not a medical device recall event is large scale, potential disruptions to healthcare supply chains can be detrimental to patients and health professionals (R. Jayaraman, AlHammadi, & Simsekler, 2018). As such, recall strategies may range from proactive to simply reactive (Hora, Bapuji, & Roth, 2011). The choice of recall strategy in this field has been mostly linked to the device classification assigned by the FDA in the premarket approval process and the severity of the adverse events reported by end users (Caceres, 2014).

There are a variety of factors that contribute to the recall decision-making process. While patient safety is at the top of the list, manufacturer liability and the financial implications of a recall event are also important influences to consider (Ball, Shah, & Donohue, 2018). While some manufacturers may choose to adopt a proactive stance to product recalls, a study conducted by Chen, Ganesan, and Liu (2009) found that the stock market rewards these approaches less than passive approaches. Other studies have shown that there is a mixed market response to mass recalls for manufacturers that provide a wide variety of products (Thirumalai & Sinha, 2011; Kini, Shenoy, & Subramaniam, 2013). Nevertheless, Redberg and Dhruva (2011) argued that despite the

uncertainty behind the actual financial consequences of recalls, consumer safety should outweigh all other factors.

2.3.5 Ventilator Recalls

Under the "ventilator" category within the FDA's medical devices databases, there are devices in all three classification levels. Because not all ventilators are classified as Class III devices (the riskiest FDA classification), certain manufacturers have adopted a reactive strategy to recalling faulty devices. For example, the FDA notified users of Vyaire Medical ventilators, a Class II medical device, that the manufacturer had recalled more than 2,600 devices due to software failures only after multiple patient injuries had already been sustained (Center for Devices and Radiological Health, 2022). Similarly, in June of 2021, Philips issued a massive voluntary recall notification for six different ventilator types manufactured between 2009 and early 2021, amounting to about 5.2 million devices distributed worldwide (Center for Devices and Radiological Health, 2021). This recall event occurred after more than 1,200 adverse event reports had been submitted complaining about malfunctioning devices and at least 100 injuries had already been sustained by patients (Park, 2022). These cases demonstrate why it is crucial for ventilator manufacturers to efficiently mine through the data contained within the MAUDE database to actively monitor what their products' end users are saying about their devices.

2.4 Use of Natural Language Processing and Text Mining for Product Defect Discovery

Due to the significant amount of effort required for information extraction from narrative data, NLP and text mining techniques have been in development since the late

1940s in an attempt to computerize the data analysis process (Liddy, 1990). From the 1960s to the 1980s, text mining methods were mostly utilized for government and life sciences applications (Hobbs, Walker, & Amsler, 1982). Since the late 1990s, however, researchers have also been applying text analytic techniques to complex business problems to reformat unstructured data sources in order to derive patterns and automatically extract necessary information (Hearst, 1999). Today, the availability of online user-generated content has led organizations to heavily invest in big data analytics methods to sift through the enormous amounts of valuable consumer feedback regarding their products (Abrahams, Fan, Wang, Zhang, & Jiao, 2015).

Through social media, for example, consumers may choose to express their opinions via videos, messages, or comments, which ultimately provides manufacturers a trove of unstructured data that are difficult to analyze (Lohr, 2012). To aid in the product defect discovery process, Abrahams, Fan, Wang, Zhang, and Jiao (2015) suggested using a text analytic framework that will help manufacturers automatically identify key warning signs of defects from social media user content. Their model applies multiple approaches, such as principal component analysis and multivariate logistic regression, to social media data to extract signals of product defects within the automotive and consumer electronics fields. Within these contexts, the researchers determined that the use of certain words, product-specific features, and semantic factors were the variables that served as the strongest indications of product defects (Abrahams, Fan, Wang, Zhang, & Jiao, 2015).

2.4.1 Review of Topic Modeling

As the main goal of this praxis is to extract early warning signs of product defects from text narratives, a topic modeling approach was employed. Information retrieval methodologies have been in development since the 1970s, when the term frequency-inverse document frequency (tf-idf) methodology was first introduced by Spärck Jones (1972). This term weighting technique calculates a specific term's frequency in an individual document within a corpus and then compares it to the term's frequency within the entire corpus. Salton and McGill (1983) incorporated this methodology to text corpora from Internet search engines by reducing them into vectors of real numbers that could be efficiently processed. This method was later improved upon by Deerwester et al. in their 1990 publication on latent semantic indexing (LSI) as an alternative approach to document dimensionality reduction. Latent semantic indexing addressed the issue of unreliable term-document association from the simple tf-idf method by treating information retrieval as a statistical problem that considered the context in which words appear (Deerwester et al., 1990). Deerwester et al. (1990) set out to resolve the issues of synonymy and polysemy that plagued the original tf-idf methodology. In 1999, the probabilistic latent semantic indexing (pLSI) method was proposed by Hofmann as a more statistically supported alternative to the original LSI method that models individual terms in a document as a sample from a mixture of "topics" by incorporating a latent class model approach. While pLSI provides a more solid probabilistic foundation to text modeling than LSI, it does not provide a document-level probabilistic model, which consequentially leads to overfitting and an inability to assign probabilities to documents external to the training set (Blei, Ng, & Jordan, 2003). To address this issue, Blei, Ng,

and Jordan (2003) proposed the LDA topic modeling approach, which considers both the exchangeability of terms and documents when developing the mixtures of "topics."

2.4.2 Topic Modeling Applications

Latent Dirichlet Allocation can be used to successfully extract hidden topics from collections of textual datasets in virtually any field of study (Jelodar et al., 2019). This topic modeling approach has been applied in a variety of fields, such as linguistic science (Lui, Lau, & Baldwin, 2014; Vulic, Smet, & Moens, 2011), political science (Greene & Cross, 2015; Cohen & Ruths, 2021), software engineering (Linstead, Rigor, Bajracharya, Lopes, & Baldi, 2007; Gethers & Poshyvanyk, 2010), and crime prediction (Gerber, 2014; Chen, Santoso, Lee, & Wang, 2015). In the medical and biomedical fields, considerable work has been conducted utilizing this methodology to extract latent topics from healthcare narrative datasets. For example, Geletta, Follett, and Laugerman (2019) applied LDA topic modeling to clinical trial data obtained from a government repository for the prediction of clinical trial failures. They applied LDA to the "unstructured" narrative data to obtain the model's generated topic groupings and compared the coherency of the words assigned to each with the "structured" labels provided for each clinical trial studied. Their study demonstrated how the use of LDA-generated topic groupings from "unstructured" data were more representative of the latent topics in the narrative text than the "structured" variables.

Studies focusing on product defect discovery have also utilized LDA to uncover defect signals from narrative data. Fong, Sarkani, and Fossaceca (2021) suggested applying a combined approach of recurrent neural networks and LDA to Amazon customer review data to extract hidden topics that may provide indication of product

defects. Their model successfully extracted a series of topics directly related to product defects from Amazon reviews that had been submitted for a set of home furnishing products. Their findings provided useful information for product manufacturers and product recall insurance teams, thereby reducing the potential for economic loss from product defects encountered by customers. This research provided a solid foundation for this praxis, as LDA was shown to successfully extract early signs of product malfunctions from narrative data.

2.4.3 Review of Text Classification

Within the fields of NLP and text mining, text classification problems have been the most widely studied for various applications (Kowsari et al., 2019). Text classification attempts to label and group together records by categories from already-defined datasets (Sebastiani, 2002). In the medical field specifically, this technique has been utilized for a variety of applications, such as automated classification of medical texts (Prabhakar & Won, 2021), classification of unstructured clinical narratives for billing purposes (Venkataraman et al., 2020), and clustering of patients with similar conditions to examine quality of care in hospitals (Cerrito, 2007).

Regardless of the application, the majority of text classification systems begin with a raw dataset and then follow the following four phases, as depicted in Figure 2-3: (1) feature extraction, (2) dimensionality reduction, (3) classification model selection, and (4) model evaluation (Kowsari et al., 2019).

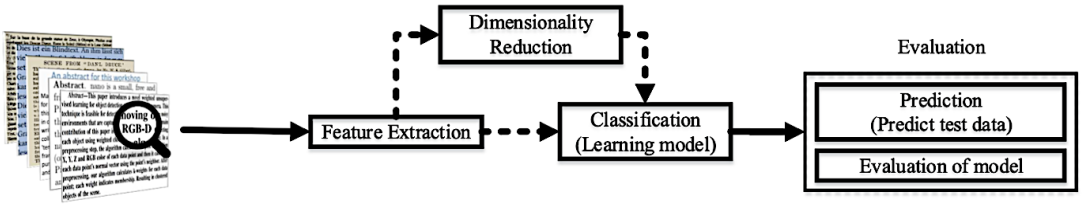

Figure 2-3. NLP Pipeline (Kowsari et al., 2019)

Prior to initiating the feature extraction step, the raw dataset is preprocessed by a series of tasks known as tokenization, stop-word removal, part-of-speech (POS) tagging, and word normalization. These preprocessing steps restructure the dataset in an attempt to increase text classifiers' accuracy and decrease the computational time required to run them (Rosid, Fitrani, Astutik, Mulloh & Gozali 2020).

- Tokenization consists of breaking down a stream of words or phrases into more meaningful elements, known as "tokens," to more efficiently calculate word occurrence in a document (Gupta & Malhotra, 2015).
- Stop-word removal is the process of removing abbreviated, irrelevant, or "noisy" words (e.g., "after," "above," "a") that do not contribute any value to the dataset itself (Saif, He, & Alani, 2018).
- POS involves classifying tokens by their corresponding part of speech based on the context in which they are being used in a sentence (Lim & Park, 2020).
- Word normalization, or the process of converting a word into its normalized form, can occur in two ways, either through the stemming of words or through lemmatization (Uysal & Gunal, 2014). For the purposes of this praxis, lemmatization was selected, as this technique provides the actual base form of a

word, or "lemma," instead of the "stem" or "generally normalized form" that is obtained via stemming (Toman, Tesar, & Jezek, 2006).

From the cleaned dataset, feature selection is conducted to determine the relevant subset of features that are most relevant to the target concept (Dash & Liu, 1997). To mitigate the impact of irrelevant features in datasets, feature selection methods are employed to maximize the model's classification accuracy and ensure that the final class distribution mimics the initial class distribution when only given information pertaining to chosen features (Dash & Liu, 1997).

The subsequent step transforms high-dimensional datasets into datasets of reduced dimensionality, while still retaining the same intrinsic dimensionality (Van Der Maaten, Postma, & Van den Herik, 2009). This step in the pipeline aids in the classification process and alleviates the negative effects of dimensionality in datasets. The most commonly used linear method of dimensionality reduction is principal component analysis (PCA). This method works by constructing a lower-dimensionality version of the dataset that describes the majority of the variance included in the data (Kowsari et al., 2019).

Prior to selecting text classification models, the data have to be prepared further by creating labels and applying numbers to these labels. As described in **Section 2.4.1**, a commonly used algorithm for this process is tf-idf, whereby a measure of originality for a word is obtained by comparing word frequency in a single record versus the number of records in which the word appears. This approach helps to minimize the effect of frequently occurring words in the dataset (Kowsari et al., 2019).

Text classification model selection is one of the most crucial steps in the pipeline, and a thorough understanding of the functionality of each algorithm is essential, as there is no "best model" option for any given dataset (Dalal & Zaveri, 2011). The text classification models explored by this praxis have been commonly employed by researchers within many fields of study for a variety of applications. **Table 2-2** contains a brief description of each, as explained by Kowsari et al. (2019) in their survey of text classification algorithms.

Classifier Model	**Description**
Logistic Regression	A statistical linear classifier that predicts data values based on past observations by examining the relationships between features and specific outcome probabilities.
Support Vector Machines	A supervised learning algorithm that finds a hyperplane or a line that linearly separates a dataset into separate classes. The goal of this classifier is to accurately classify new data by selecting the hyperplane that maximizes the margin, or the distance between the nearest point of all classes to the line.
Naïve Bayes	A classifier based on Bayes' theorem that assumes independence among features, given the class. Classification prediction is achieved by choosing the highest posterior probability of a record belonging to any given class.
Stochastic Gradient Descent	A linear classifier, such as SVM, with SGD training that finds the gradient of the hinge loss function to better estimate the SVM hyperplane and provide more accurate classification results.

Random Forest	A supervised learning model that makes a set of decision trees from a randomly chosen subset of the dataset. It then tabulates the final classification "votes" of each decision tree to make the final class prediction.

Table 2-2. Text Classification Models (Kowsari et al., 2019)

In addition to these models, techniques have also been proposed that combine predictions from multiple trained models to increase prediction performance. Sagi and Rokach (2018) advocated for ensemble learning techniques as state-of-the-art approaches to machine learning problems. One of the most popular ensemble learning techniques is known as the AdaBoost algorithm, which combines multiple "weak" learners by sequentially adjusting the weights of previously misclassified instances, therefore "growing" subsequent learners from prior ones (Oza & Tumer, 2008). Due to the complex nature of real-world data, ensemble learning methods have been the preferred method for data classifier applications, as they are capable of providing solutions that could otherwise never be achieved by a single algorithm. Successful applications of ensemble methods have been achieved in a variety of industries, such as remote sensing (Giacinto & Roli, 1997), person identification (Erdoğan, et al.,2005), and medicine (Svetnik, Liaw, Tong, & Wang, 2004).

The final step in the text classification pipeline is to evaluate the predictive performance of the algorithms under consideration to select the best model. While there is much debate about the ability of current performance metrics to accurately generalize within different contexts, the basic functionality of some commonly used metrics—such as model accuracy, sensitivity, specificity, precision, recall, and F1 score—are still the

preferred method of model comparison (Japkowicz, 2006). These metrics, based on the "confusion matrix" shown in **Figure 2-4**, measure the relationship of predicted true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). **Table 2-3**, adapted from Kowsari et al. (2009), presents the mathematical computations associated with each measure.

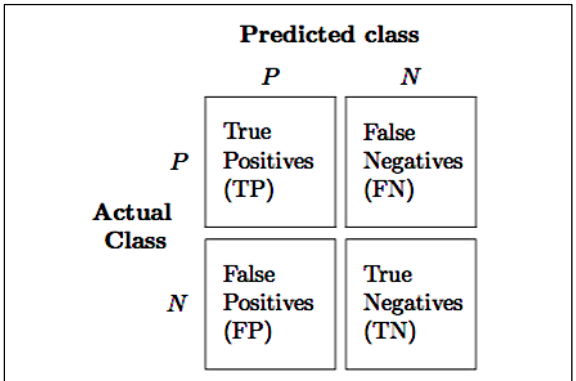

Figure 2-4. 2x2 Confusion Matrix

accuracy	$\frac{(TP+TN)}{(TP+FP+FN+TN)}$
sensitivity	$\frac{TP}{(TP+FN)}$
specificity	$\frac{TN}{(TN+FP)}$
precision	$\frac{\sum_{l=1}^{L} TP_l}{\sum_{l=1}^{L} TP_l + FP_l}$
recall	$\frac{\sum_{l=1}^{L} TP_l}{\sum_{l=1}^{L} TP_l + FN_l}$
$F1 - Score$	$\frac{\sum_{l=1}^{L} 2TP_l}{\sum_{l=1}^{L} 2TP_l + FP_l + FN_l}$

Table 2-3. Model Performance Metrics (Kowsari et al., 2009)

2.4.4 Predictive Modeling Applications

Text classifiers have been extensively utilized for the identification of product recalls. Bleaney, Kuzyk, Man, Mayanloo, and Tizhoosh (2018) developed an automatic classification system to detect safety issues in children's products with a precision metric that outperformed the existing models on the subject at the time when their study was published. Through experimentation with various classifiers—such as Logistic Regression, Support Vector Machines, Naïve Bayes, Random Forests, and ensemble methods—their highest-performing model achieved 66% precision, an improvement from the prior Consumer Product Safety Commission benchmark of 50% (Bleaney, Kuzyk, Man, Mayanloo, & Tizhoosh, 2018).

Mukherjee and Sinha (2018) proposed an integrative model to evaluate judgement bias in manufacturer decisions involving recalls for a product mix of medical devices. Despite their focus being on judgement bias evaluation, their methodology included predictive modeling in their framework to ascertain recall likelihood from adverse event reports included in the MAUDE database. Their research was able to predict manufacturer recall decisions using a variety of variables, such as specific product data (device age, FDA risk level classification, whether or not it is an implantable device, and product usage), manufacturer-related features (e.g., firm size, product portfolio index, fixed assets, and mergers and acquisitions investments), and number of competing product models (Mukherjee & Sinha, 2018). However, no consideration was given to product defects or early warning signs of product malfunctions, a gap addressed by this praxis in the context of ventilators.

2.5 Summary and Conclusion

The literature review presented in this chapter provided an extensive analysis of existing work regarding the medical device industry and the medical device design process, medical device defects and recalls, and NLP and its applications within diverse fields of study. Existing studies related to ventilators demonstrate how the complexity of these devices leaves room for further analysis. An emphasis was placed on LDA topic modeling and text classifier models as prediction tools for product defect identification. After this extensive literature review was performed, it was noted that few scholars have developed recall prediction models for medical devices using adverse event reports from the MAUDE database. Furthermore, no studies were found that focused on defect detection for ventilators using these tools. To fill this gap, this praxis proposes a combination of LDA and predictive modeling techniques to extract early warning signs of ventilator defects and predict potential recalls.

Chapter 3—Methodology

3.1 Introduction

This praxis focuses on the ability of a variety of text mining applications to (1) extract early warning signs of ventilator defects through topic modeling, (2) predict the recall status of these devices utilizing the identified warning signs as feature vectors in text classifiers, and (3) predict the recall classification level by using textual data as feature vectors in text classifiers. The initial dataset, sourced from the Food and Drug Administration's (FDA) Manufacturer and User Facility Device Experience Database (MAUDE) and the Medical Device Recall database, underwent an extensive preprocessing phase, and the "clean" dataset was then analyzed by the various machine learning algorithms.

The topic modeling approach utilized to identify early warning signs of ventilator defects from the narrative reports is Latent Dirichlet Allocation (LDA). Following the same LDA results validation strategy as Fong, Sarkani, and Fossaceca (2021), this study employed a group of subject matter experts (SMEs) to validate if the extracted topics from the LDA model were considered early warning signs of ventilator defects. The trained LDA model and the resulting early warning signs were then utilized as feature vectors in text classifiers to predict recall status. Finally, the recall classification level is predicted from the narrative data as an additional step in the recall management process.

After the literature review was conducted as described in **Chapter 2**, it was discovered that several text classifiers have proven effective when used in the context of

product defect detection (Bleaney, Kuzyk, Man, Mayanloo, & Tizhoosh, 2018). The models included were Logistic Regression, Support Vector Machines, Naïve Bayes, Random Forests and ensemble methods. Due to the promising results obtained by these models, as discussed in **Section 2.4.4**, the models employed for the predictive modeling portions of this praxis include Stochastic Gradient Descent (SGD) used to train a Support Vector Machine (SVM) classifier, Random Forest (RF), AdaBoost (AB), Gaussian Naïve Bayes (NB), and Logistic Regression (LR). **Figure 3-1** details the end-to-end process explored in this praxis and is referenced throughout the chapter.

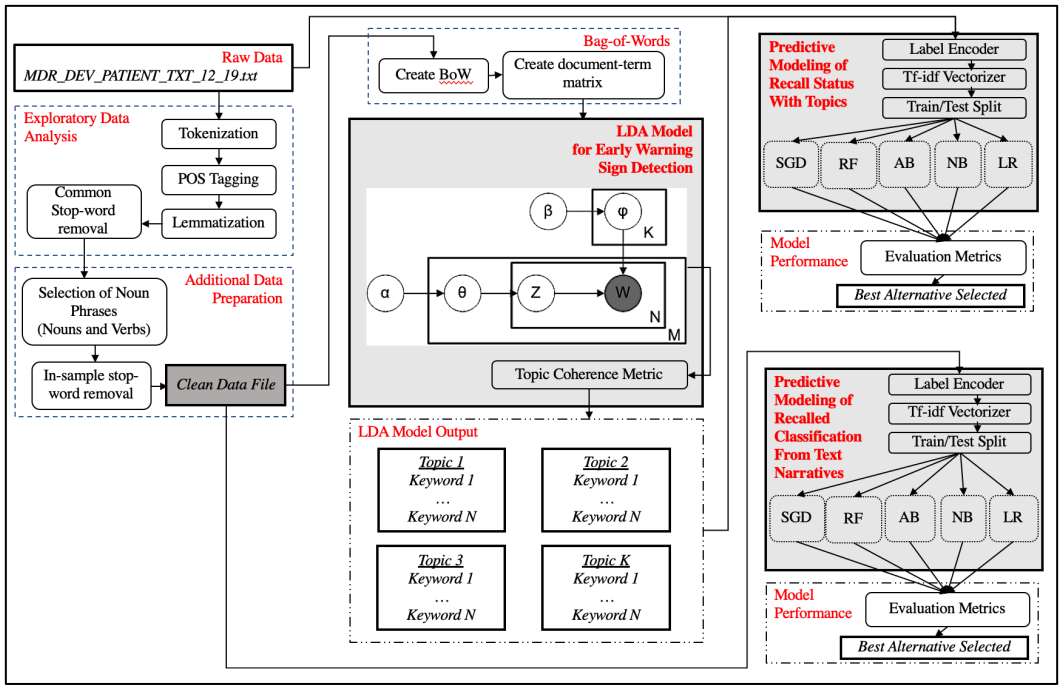

Figure 3-1. Methodology End-to-End Process

3.2 Data Collection and Data Sources

The data utilized for this praxis were sourced from the FDA's MAUDE database and the Medical Device Recall database. Adverse event narrative reports from the

MAUDE database were used in conjunction with medical device recalls to identify which medical devices have or have not been recalled from the market. The data sourced concerns 94 ventilation devices included in the product code classifications shown in **Table 3-1**. These product codes were chosen because they represent a mix of all three FDA device risk classifications (Class I, Class II, and Class III) and contain both life-sustaining and non-life-sustaining devices.

Product Code	Device Type	FDA Device Risk Classification	Life-Sustaining Flag
BSZ	Gas-Machine, Anesthesia	2	Y
BTM	Ventilator, Emergency, Manual (Resuscitator)	2	Y
BZD	Ventilator, Non-Continuous (Respirator)	2	N
BZO	Set, Tubing And Support, Ventilator (W Harness)	1	N
CBK	Ventilator, Continuous, Facility Use	2	Y
LSZ	Ventilator, High Frequency	3	N
NOU	Continuous, Ventilator, Home Use	2	Y

Table 3-1. Summary of Product Code Classifications, Device Types, Associated FDA Device Risk Classification, and Life-Sustaining Flag (sourced from Center for Devices and Radiological Health, n.d.)

The data collection process was divided into three main phases: (1) downloading and linking the raw data files from the MAUDE native data file collections, (2) filtering

for relevant records for this praxis, and (3) linking the recall status data to the MAUDE data file. The MAUDE datasets included on the FDA website contain numerous downloadable files, with historical records dating back to 1991 for all medical devices regulated by the FDA. As of March 1st, 2022, there are 68 datasets included in the MAUDE database, which are updated weekly and formatted as pipe-delimited text files. The datasets are organized by the information contained (i.e., master data, patient problem data, device data, narrative data) and the time period covered; therefore, individual files may contain anywhere from ~990 to ~13 million entries. **Appendix A** contains the full list of all downloadable files currently found in the MAUDE database. To limit the scope of this research to relevant events, the sourced data included files in the database from 2012 to 2019.

3.2.1 Importing and Linking MAUDE Data Files

The MAUDE database houses data files grouped into four main types (master, service, patient, and text). All data files are available as zipped files, are pipe-delimited, and only include one record per line. Header rows are included for all file types, which are denoted through this praxis in all capital letters. A summary of each of the files downloaded is provided in **Figure 3-2**.

File Type	Zipped Text File Name	Unique Identifiers	File Description
Master	*mdrfoithru2020*	*Uniquely identified by MDR_REPORT_KEY*	Master Record Data
Device	*device2012*	*Uniquely identified by combining MDR_REPORT_KEY and DEVICE_SEQUENCE_NO*	Device Problem Data
	device2013		
	device 2014		
	device2015		
	device2016		
	device2017		
	device2018		
	device2019		
Patient	*patientthru2020*	*Uniquely identified by combining MDR_REPORT_KEY and PATIENT_SEQUENCE_NO*	Patient Record Data
Text	*foitext2012*	*Uniquely identified by MDR_TEXT_KEY or by combining MDR_REPORT_KEY, PATIENT_SEQUENCE_NO and TEXT_TYPE_CODE*	Adverse Event Narrative Report Data
	foitext2013		
	foitext2014		
	foitext2015		
	foitext2016		
	foitext2017		
	foitext2018		
	foitext2019		

Figure 3-2. MAUDE File Type Descriptions

Since the master and patient data files contained all records dating from 1991 to 2020, the records were filtered to only include data from 2012 to 2019. As the device files and text files were separated by year, these files were run through a Python code that concatenated them into a single file and deleted duplicate records containing the same MDR_REPORT_KEY. **Appendix B** contains the code used to achieve this result for the text files. The code was then reused for the device files.

All filtered files were then downloaded into Microsoft Access and linked. The MDR_REPORT_KEY, PATIENT_SEQUENCE_NO, and the DEVICE_SEQUENCE_NO fields were used to uniquely identify events and link the data files together. As each file type has unique identifiers and not all are related to each other, the linking process was divided into various steps. **Figure 3-3** illustrates the relationships that were used to link the records in each of the imported files.

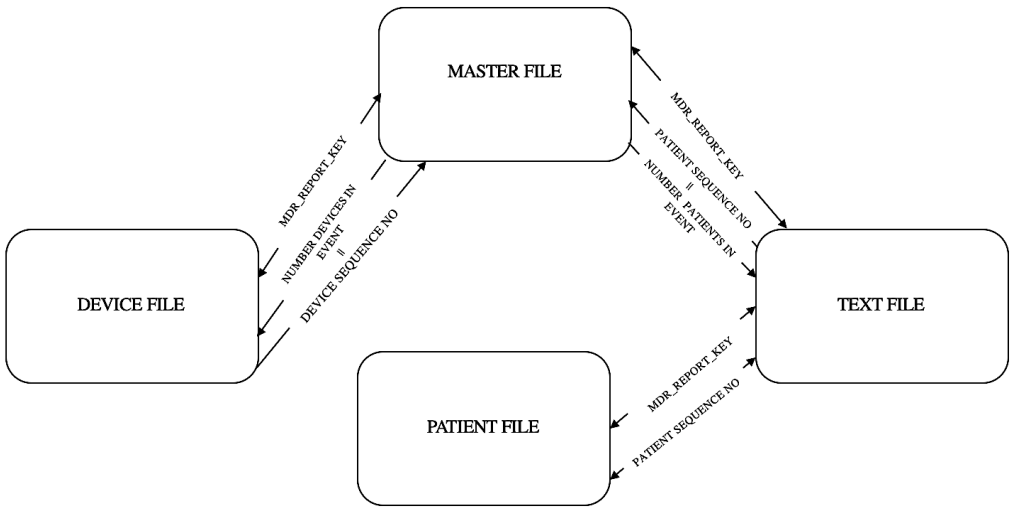

Figure 3-3. MAUDE File Type Relationships Used to Link Records

As most of the information contained in the adverse event reports is free-text entry, there are inconsistencies in the way the information is presented. **Figure 3-4** depicts one example from the raw dataset in which the MANUFACTURER_NAME, BRAND_NAME, and MODEL_NUMBER fields referenced the same ventilator model, but were not consistently reported. While the MANUFACTURER_NAME and BRAND_NAME fields are consistent, the MODEL_NUMBER field contains typos that required further data cleaning, as the correct entry should have simply been "980."

MDR_REPORT_KEY	DATE_OF_EVENT	EVENT_TYPE	MANUFACTURER_NAME	BRAND_NAME	MODEL_NUMBER
8020893-2014-1955	7/16/14	Injury	PURITAN BENNETT CORP	980 VENTILATOR	980 VENTILAOTR
8020893-2018-235	5/4/18	Malfunction	PURITAN BENNETT CORP	980 VENTILATOR	980
9207706	6/21/19	Malfunction	PURITAN BENNETT CORP	980 VENTILATOR	PURITAN BENNETT PB 980 VENTILATOR SYSTEM

Figure 3-4. Raw Data Example of Inconsistent Model Number Labeling

Once all typos were corrected to achieve consistency across the MANUFACTURER_NAME, BRAND_NAME, and MODEL_NUMBER fields, the dataset was ready to be matched with the appropriate recall status and recall classification level information.

3.2.2 Medical Device Recall Data

The recall data were obtained from the FDA's Medical Device Recall database. These data were collected by searching the database for each of the product codes (BSZ, BTM, BZD, BZO, CBK, LSZ, and NOU) listed in **Table 3-1** to obtain information about recalled devices in these categories from January 1st, 2012 to December 31st, 2019. As each downloaded file contains the same 11 columns, shown in **Figure 3-5**, all files were concatenated into a single Excel file to facilitate further analysis.

WEB ADDRESS	RECALL NUMBER	PRODUCT DESCRIPTION	TRADE NAME	RECALL CLASS	CENTER	CENTER CLASSIFICATION DATE	POSTED INTERNET DATE	TERMINATION DATE	FIRM NAME	MANUFACTURER RECALL REASON

Figure 3-5. Recalled Medical Devices Data Structure

For the purposes of this study, the only relevant fields considered were PRODUCT DESCRIPTION, TRADE NAME, RECALL CLASS, CENTER CLASSIFICATION DATE, and FIRM NAME. All remaining columns were removed from the dataset.

3.2.3 Dataset Structure

In order to match recall data to adverse event reports, relationships had to be found within the datasets, as the columns were not uniformly labeled. The MANUFACTURER_NAME field was linked to the FIRM NAME field, and the

BRAND_NAME and MODEL_NUMBER fields were used to link recall status and the classification level of records using information contained in both the TRADE NAME and PRODUCT DESCRIPTION fields. For specific ventilator models that were not found in the recall dataset, additional research was conducted by searching through manufacturer websites and medical journals to determine whether or not those devices had been recalled in the period under review. If no information was found, the MODEL_NUMBER referenced by the record was determined to have not been recalled.

After the process of matching ventilator adverse event reports to the applicable recall status was completed, the datasets were consolidated into one Excel file, titled "MDR_DEV_PATIENT_TXT_12_19." The final dataset's structure was modeled using the columns shown in **Table 3-2**. This process was accomplished by removing columns that would not contribute relevant information for the purposes of this study. The removed columns included the WEB_ADDRESS, RECALL_NUMBER, CENTER, and MANUFACTURER RECALL REASON fields, which would have inserted noise into the dataset. This praxis mainly focuses on the analysis of the textual data found in the EVENT_TEXT column.

Column Header	Column Description
MDR_REPORT_KEY	Unique report number identifier of the adverse event record
DATE_OF_EVENT	Date of event referenced in report

EVENT_TYPE	Type of event being reported (D = Death, IN = Injury, IL = Injury, IJ = Injury, M = Malfunction, O = Other)
MANUFACTURER_NAME	Name of device manufacturer
BRAND_NAME	Device identification information
MODEL_NUMBER	Device identification information
DEVICE_REPORT_PRODUCT_CODE	Device classification product code used to group similar devices together
DATE_RECEIVED	Date report was received
RECALL_STATUS	Was device recalled? (Y = Yes, N= No)
RECALL_STATUS_CLASSIFICATION	Recall status classification (N/A = Not Recalled, 1 = Class I, 2 = Class II, 3 = Class III)
EVENT_TEXT	Adverse event text narrative report

Table 3-2. Dataset Structure

The literature review conducted for this study revealed that the matching scheme performed here was also adopted by Ensign and Cohen (2017) and Khanal (2018) when sourcing FDA MAUDE data for analysis. In addition, as each record had to be checked individually to ascertain the correct recall classification, SMEs in the fields of medical device design and machine learning were consulted and their guidance was followed for the process of record matching. Following previous studies (Ensign & Cohen, 2017; Fong, Sarkani, & Fossaceca, 2021), this praxis also utilized SMEs for data and results validation at various stages. The medical device design engineers' experience averaged at 12 years, and one was a current FDA-certified medical device expert. The machine

learning experience among SMEs averaged at 10 years, mostly spent in the data science architecture field, with special emphasis on Python data analytics and topic modeling.

3.3 Dataset Preprocessing

Once unique records were identified and appropriately linked to all necessary fields, the raw text data were preprocessed into readable information that could be interpreted by machine learning software. The dataset was rechecked for duplicate entries, and only the first record among duplicates was kept to reduce noise in the dataset that may lead to skewed results. Following this step, the dataset was cleaned by removing records with missing fields in the columns of interest.

3.3.1 Exploratory Data Analysis and Text Preprocessing

Text preprocessing is an essential part of the NLP pipeline, as it mitigates the effect of human-introduced errors in a dataset (Zheng, He & He, 2020). The activities performed within this phase of the pipeline are depicted in the *Exploratory Data Analysis* and *Additional Data Preparation* sections of the end-to-end process presented in **Figure 3-1**. As mentioned in **Section 2.4.3**, the steps taken in this study to improve the quality of the text data involved (1) POS tagging, (2) stop-word removal, and (3) word normalization. These tasks were accomplished by utilizing Python Spacy libraries. **Appendix C** contains a sample of the code utilized to achieve these preprocessing steps.

Part-of-speech tagging is an NLP technique that classifies words by their part of speech (e.g., noun, verb, adjective) so that text classifiers can better understand the context in which the words are used in a sentence (Lim & Park, 2020). For this study, the

frequencies of POS tags were tabulated and sorted from highest occurrence to lowest to determine which features to use for the classification models. The code utilized for this step is included in **Appendix C-1**. As there was a large number of nouns and verbs in this particular dataset, the POS features chosen for modeling were a combination of nouns, verbs, and noun phrases. This combination of nouns and verbs as POS features has been shown to improve the quality of text classifier models. Al-Omari (2019) successfully implemented this POS tagging approach and utilized the extracted nouns and verbs from nine separate datasets for text classification applications. The three highest classification accuracies achieved by the algorithms employed—k-nearest neighbors, Naïve Bayes, and Support Vector Machines—were 86.83%, 93%, and 72.41%, respectively (Al-Omari, 2019).

The selected features then underwent a lemmatization step in which the words were reduced to their basic word forms, or lemmas, by replacing their suffix with another or removing it entirely (Toman, Tesar & Jezek, 2006). As discussed by Toman, Tesar, and Jezek (2006), this method is preferred over the alternative stemming technique, as it is generally better suited for classification processes. The code utilized for this step is included in **Appendix C-2**.

Stop-word removal was then employed to eliminate words from the text data that are of little value to the overall context of the dataset. This preprocessing step ultimately allows for reduced dimensionality of the feature space, thus improving the overall classification accuracy of the model (Uysal & Gunal, 2014). The code utilized for this step is included in **Appendix C-3**. A sample of a raw adverse event narrative report as well as its preprocessed version are shown in **Figure 3-6** and **Figure 3-7**, respectively.

```
Event Description: THE CUSTOMER REPORTED THAT THE 840 VENTILATOR STOPPED CYCLING WHILE IN USE ON A P
T. THE PT WAS NOT HARMED OR INJURED AS A RESULT OF THE EVENT. THE CUSTOMER REPORTED TO HAVE NOT DUPL
ICATED THE ALLEGED MALFUNCTION. THE VENTILATOR PASSED ALL TESTING.
```

Figure 3-6. Raw Adverse Event Narrative Text Report

```
['event', 'description', 'customer', 'report', 'stopped', 'cycle', 'use', 'harmed', 'result', 'event
', 'customer', 'report', 'duplicate', 'malfunction', 'ventilator', 'pass', 'testing', '840_ventilato
r', 'alleged_malfunction']
```

Figure 3-7. Preprocessed Version of an Adverse Event Narrative Text Report

As an additional data cleaning step prior to classification, the dimension of the dictionary generated by the lemmatization process was studied to verify the amount of unique lemmas that would be present in the data. This extra step enabled the identification of potential corpus-related stop-words that needed to be deleted from the dataset to further clean it from noisy data that did not add any value and increased the models' processing times (Pesaranghader, Pesaranghader & Rezaei, 2013). Certain lemmas, such as "event" and "description," were found in each of the adverse event narratives, as shown in **Figure 3-7**. As these words do not add value to the information contained in the record, they were considered noise in the dataset. To reduce the dimensionality of the dictionary for classification and topic modeling, lemmas with a frequency count lower than two records or those that appeared in 99.9% of records were deleted (i.e., "event" and "description"). The code utilized for this step is included in **Appendix C-4**.

The final features selected for the models were a combination of adverse event narrative report text length and the list of identified lemmas from the dataset. The final preprocessed dataset, referenced in **Figure 3-1** as "Clean Data File," was coded in a separate notebook titled "clean_data.pickle" and saved in a prespecified output folder. By saving the file in this format, it was easy to locate for subsequent use in other models and the file preserved its structure. The code utilized for this step is included in **Appendix C-5**.

3.4 Model Development

This praxis explores three main machine learning applications, separated into three phases, to assist ventilator manufacturers in the identification of early warning signs of defects and potential recalls. The first phase employs the LDA topic modeling technique to extract latent topics from the adverse event data. The 10 most salient terms for each topic were evaluated to determine if they could be considered early warning signs of ventilator defects that may lead to a recall. The LDA model then assigned a topic distribution to each record, along with the probability of belonging to each assigned topic. The second phase of this praxis utilized the topic distributions obtained for each record as feature vectors to predict recall status. For the third phase, text classification was performed to obtain the associated recall classification level of recalled devices, utilizing the narrative data from the EVENT_TEXT column. The machine learning algorithms employed for phase two and three were SGD used to train a SVM classifier, RF, AB, NB, and LR.

3.5 Latent Dirichlet Allocation Topic Model for Early Warning Detection of Ventilator Defects

The process of topic modeling essentially uncovers latent, or hidden, topics found in unstructured textual document sets in order to group them by common themes (Zhao et. al., 2015). Applying LDA topic modeling to online customer reviews is a common method to provide engineers a gateway into the detection of keywords associated with product defects and even potential resolutions to issues found in products (Fong, Sarkani, & Fossaceca, 2021). For the purposes of this praxis, LDA topic modeling was applied to the EVENT_TEXT column included in the preprocessed dataset to assist manufacturers in the process of uncovering early warning signs of ventilator defects. This phase was presented in the *Bag-of-Words*, *LDA Model for Early Warning Sign Detection,* and *LDA Model Output* sections of the end-to-end process depicted in **Figure 3-1**. The preprocessed *Clean Data File* block in this depiction is connected to the *Bag-of-Words* section, which begins the LDA topic modeling process.

In general, the LDA topic modeling process begins with building a probabilistic text model by making the assumption that a record was generated by selecting a set of topics, k, and then a set of words for each topic (Blei, Ng, & Jordan, 2003). Latent Dirichlet Allocation then randomly distributes k topics across each record, m, by assigning a topic to each word w, denoted as α. The process of randomly assigning topics to words in each record creates α, the per-record topic distributions θ on m records, and β, the per-topic word distribution ϕ on k topics. For each of the words w in record m, LDA assumes the topic assignment is wrong, while the rest of the topic assignments are

correct. It then probabilistically reassigns words *w* to topics *z* based on θ, or which topics *k* are in record *m*, and ϕ, how many times word *w* has been assigned to topic *z* over all records, *m*. This process is continued iteratively until the model converges with *k* number of topics, each having word distribution ϕ that best fits each record *m* (Blei, Ng, & Jordan, 2003). The collection of words *w* included in each ϕ is used as the key signals needed to detect product defects. The plate notation for the LDA topic model is presented in **Figure 3-8**.

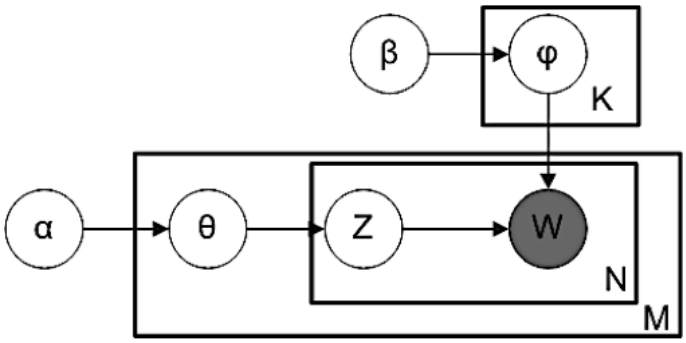

Figure 3-6. Plate Notation of LDA Topic Model (Blei, Ng & Jordan, 2003)

For this praxis, the variables in the LDA model are defined as follows:

- α = per-record topic distributions,
- β = per-topic word distribution,
- θ = topic distribution for record *m*,
- ϕ = word distribution for topic *k*,
- z = topic for the *n*-th word in record *m*, and
- w = specific word.

Latent Dirichlet Allocation provides a list of topics extracted from the dataset, but cannot determine labels for the model to test whether the labels are accurately assigned (Fong, Sarkani, & Fossaceca, 2021). Because of this limitation, the topic coherence metric was developed as an evaluation metric to assess the performance of an LDA topic model by measuring the semantic similarity among words appearing in the word distribution ϕ for a given topic *k* (Sbalchiero & Eder, 2020).

As described by DiMaggio, Nag, and Blei (2013), there are three ways to validate an LDA model's results: (1) testing model assumptions via statistical validation (Mimno & Blei, 2011), (2) testing model results via expert reasoning through semantic validation (Grimmer & Stewart, 2013), or (3) correlating external events with model results through predictive/external validation (Grimmer & Stewart, 2013). This praxis adopted a semantic validation scheme, similar to Asmussen and Møller's (2019) and Chang et al.'s (2009) reasoning, who also argued that human judgement is an effective way of measuring an LDA model's performance. Therefore, to validate the results of this LDA model—and following the approach taken by Fong, Sarkani, and Fossaceca (2021)—SMEs were asked to determine if the top 10 salient terms for each topic grouping were considered to be early warning signs of ventilator defects.

For this praxis, the LDA topic model was run through Python code, which can be found in **Appendix D**. The code was broken up by sections, with detailed commentary to facilitate reproducibility (all statements in the code starting with "#" explain the LDA model's functionality step by step). This process was initiated by creating a bag of words from the clean data file such that all of the identified lemmas in the dataset were grouped together. The model then counted the frequency of each lemma *w* in each narrative report

m to create the record-word matrix α. The number of topics was changed iteratively using the topic coherence metric and the cluster of words assigned to each was analyzed. Interactive visualization tools, such as the pyLDAvis plot in Python, were utilized to better understand the relationships between topics and the words assigned to each. The final output of the LDA model was evaluated using the number of topics identified and the semantic coherence of the words assigned to each. The model then added columns to the initial dataset detailing the topic distribution assigned to each record and the probability each topic had of actually belonging to the record. This trained LDA model was then saved in a prespecified output folder for use in the second phase of this praxis.

3.6 Predictive Modeling of Ventilator Recall Status from Identified Early Warning Signs of Defects

Once the LDA model extracted latent topics from the adverse event narrative reports and the results were validated as early warning signs of ventilator defects, these topics were further explored with regards to their usefulness in ventilator recall detection models. This exploration was conducted by using the topics uncovered in the LDA model as the feature vectors in the dataset. The steps performed within this phase of the pipeline are depicted in the *Predictive Modeling of Recall Status With Topics* and *Model Performance* sections of the end-to-end process shown in **Figure 3-1**. The incoming arrows into these sections emanate from the *LDA Model Output* section and the *Raw Data* block, as these are the two inputs utilized for this predictive modeling phase of the praxis. This phase was run using a Python code script, which can be found in **Appendix E**. The code was broken up by sections, with detailed commentary to facilitate

reproducibility (all statements starting with "#" explain the code's functionality step by step).

3.6.1 Data Preparation for Predictive Modeling of Recall Status Classification

This classification phase was designed to allow for the use of the saved, pre-trained LDA model on any raw dataset that has been formatted following the dataset structure shown in **Table 3-2**. However, for the purposes of this praxis, the same raw dataset used for LDA topic modeling was used for classification. For this reason, the raw dataset underwent a preprocessing step once more before being subject to text classification. This step was described in **Section 3.3.1**. Following this step, the saved, pretrained LDA model from the first phase was downloaded and the preprocessed dataset was run through the LDA model again to add the columns referencing the identified topics. Similar to the LDA model's output, the additional columns included the topics identified in each record and their associated topic distributions.

The dataset was subsequently prepared for text classification. To do so, feature vectors were defined as individual adverse event narrative report text lengths, topics identified, and their distributions for each record. Classification labels were then assigned to each category of interest, recalled or not recalled, as presented in **Table 3-3**, with each label converted into a number (0,1).

Label Number	RECALL_STATUS
0	NOT RECALLED
1	RECALLED

Table 3-3. Defined Labels for Recall Status

The dataset was then subject to a stratified sampling process to maintain a balanced proportion of label representation in the training and testing subsets. This balance allowed the models to provide unbiased predictions when faced with unseen data (Vabalas, Gowen, Poliakoff, & Casson, 2019). For the purposes of this praxis, 80% of the data were used for training/validation, and 20% of the data were set aside for testing purposes. This percentage was adopted following the guidance of Gupta, Sharma, Choudhary, and Pandey (2021), who found that this split provided the best results in classification models modeled in Python and using a dichotomous set of classes, as was the case in this phase of the praxis. A stratified k-fold cross-validation approach was utilized to validate the models' performance, with a value of k determined from running the models at 5, 10, 15, and 20 folds and comparing their results. The label proportions were then checked in the training dataset and the label containing the second smallest number of records was used as the baseline sample size for all labels with a higher number of records.

3.6.2 Classification Algorithms and Associated Parameters for Predictive Modeling of Ventilator Recall Status

The Scikit-learn library, regarded as the most robust library for machine learning in Python (Pedregosa et al., 2011), was used to design the code for this praxis. This library provides a set of parameter options for various machine learning algorithms to be used for classification. All of the available options were included in the code, as Python performed an exhaustive cross-validation grid search, known as "GridSearchCV," over all of them to find the best combination of parameter values (Pedregosa et al., 2011). The classification models were then trained using the parameters defined in **Table 3-4**.

Text Classification Algorithm	Defined Parameters
Stochastic Gradient Descent	loss = ['log','hinge'], penalty = ['l2'], alpha = [1e-6, 1e-3, 1e-1, 1e0], max_iter = [5, 1000, 10000], tol = [None, 1e-3], random_state = [3])
Random Forest	bootstrap = [True, False], max_depth = [10, 50, 100, None], max_features = ['auto'], min_samples_leaf = [1, 2, 4], n_estimators = [500, 1000], random_state = [3])

AdaBoost	base_estimator = [DecisionTreeClassifier(max_depth=1), DecisionTreeClassifier(max_depth=2)], learning_rate = [1, 1.5, 2], n_estimators = [100, 200], algorithm = ["SAMME", "SAMME.R"], random_state = [3])
Gaussian Naïve Bayes	N/A—Default parameters used
Logistic Regression	N/A—Default parameters used (L2 Regularization)

Table 3-4. Classification Algorithms and Associated Parameters for Predictive Modeling of Ventilator Recall Status

The end goal of this phase of the praxis was to select the algorithm that best predicts the associated RECALL_STATUS class for each record using the topic distributions of early warning signs of ventilator defects obtained from the LDA model. Adapted from Diab (2018) for the purposes of the classification task at hand, the problem can be described mathematically as follows:

a) *Let D_n: set of n records =$\{D_1, D_2, ..., D_n\}$.*

b) *Let C_n: set of labels that differentiate each class of the record (D), C_i=$\{C_1, C_2, C_3, ..., C_i\}$, where i is the total number of labels.*

The problem can thus be defined as needing to find the correct class (C_i) for each record (D_n) by using a classifier (f).

3.6.3 Stochastic Gradient Descent for Predictive Modeling of Ventilator Recall Status

Bottou (2010) recommended the use of SGD as a predictive modeling tool for classification problems with large-scale data. For SGD, the classifier works similarly to gradient descent, where the overall goal is to minimize the cost of inaccurate predictions by using different coefficient values for (*f*). Gradient descent is performed by comparing predicted results with actual values to choose the parameters that minimize the cost of inaccurate predictions. Stochastic Gradient Descent performs similarly, but updates the loss function parameters for each record (D_n) one by one at the end of each iteration, rather than over all samples of the training set (Bottou, 2010). The gradient descent continues down the slope of the loss function in a step fashion, otherwise known as the learning rate of the function, which is typically defaulted at 0.01 (Diab, 2018).

Table 3-4 details the type of loss function used (*log*, indicating a LR classifier with SGD training, and *hinge*, which indicates a SVM classifier with SGD training), the penalty or regularization term to be used (only allowed to run at *l2* since *l1* may eliminate features and increase computational time), alpha (*1e-6, 1e-3, 1e-1, 1e0*; constant value that multiplies the regularization term), the maximum number of passes over training data or epochs (*5; 1,000; 10,000*; to test various parameter settings), the tolerance or the stopping criteria for the algorithm (*None, 1e-3*; to test various parameter settings), and the random state (*3*, pseudo-random number set to achieve reproducible results on each run of the classifiers).

3.6.4 Random Forest for Predictive Modeling of Ventilator Recall Status

Following SGD, a RF algorithm was also explored in this phase, as scholars have suggested that RF models provide accurate results while reducing overfitting (Breiman, Last, & Rice, 2001; Pal, 2005). This method can be simply described as an ensemble of decision trees that predict each record's (D_n) class (C_i) by taking a "majority vote" on the predictions made by each individual tree (Pal, 2005). **Figure 3-9**—adapted from Sapountzoglou, Lago, and Raison (2020)—demonstrates the process by which a RF classifier makes final predictions through bagging ensemble methods.

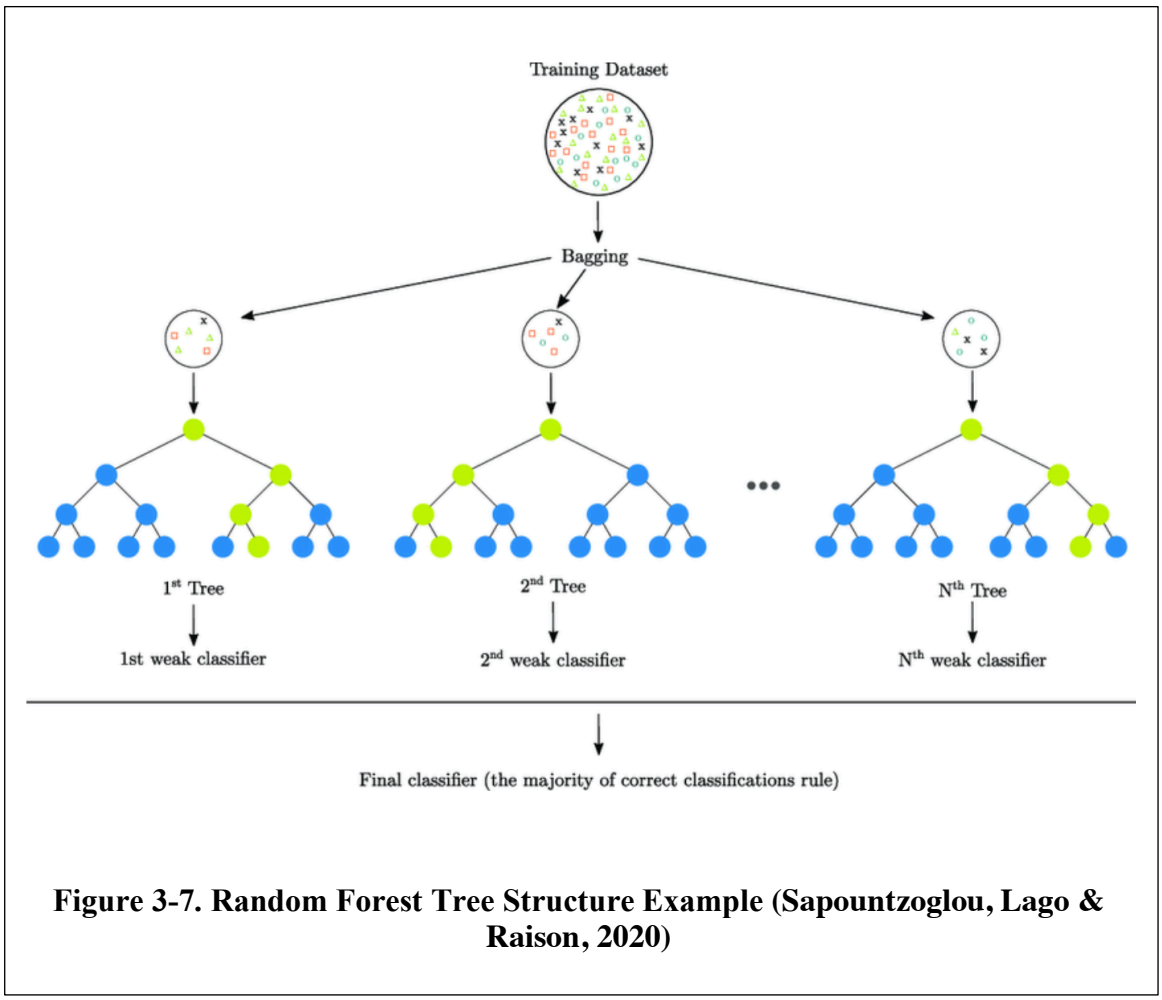

Figure 3-7. Random Forest Tree Structure Example (Sapountzoglou, Lago & Raison, 2020)

As defined by Breiman, Last, and Rice (2001), each tree classifier is created by an independently sampled random vector from the input vector. Tree induction is performed by a frequently used metric known as the Gini index for attribute selection, which calculates an attribute's impurity with respect to the classes (Breiman, Last, & Rice, 2001). For a training set, T, selecting one record at random and assigning it to some class, C_i, the Gini index is calculated as follows, where $f(C_i, T)/|T|$ is the probability that the selected record belongs to class C_i:

$$\sum \sum_{j \neq i} \left(\frac{f(C_i,T)}{|T|}\right)\left(\frac{f(C_j,T)}{|T|}\right) \quad (1)$$

Each tree is then grown to the maximum depth specified in the parameters of the model using new training data. The number of trees and features to be included in each of the trees' nodes is also determined by the parameters established in the model development process. Once all trees are built and trained, the majority vote is selected by the RF as the final class prediction for the selected record (Pal, 2005).

Table 3-4 includes parameter information pertaining to whether or not bootstrap samples were used (*True, False;* to test various settings), the maximum depth/levels of each tree generated (*10, 50, 100, None;* to test various settings), the maximum number of features to consider when choosing the best split (*"auto,"* default setting where the maximum number of features equals the square root of the number of features), the minimum number of samples required before splitting an internal node (*1, 2, 4*; default is 2, but set others to test various parameters), the number of estimators/trees in the forest (*500; 1,000*; set to start at 500 and stop at 1,000), and the random state (*3*, pseudo-random number set to achieve reproducible results on each run of the classifiers).

3.6.5 AdaBoost for Predictive Modeling of Ventilator Recall Status

An additional ensemble method explored by this study is the AB text classification algorithm using decision trees as the weak learners that need a "boost" in performance. In this case, AB works by taking the decision trees and invoking them repeatedly with different training subsets of the data, maintaining a set of weights on the original training set that can only be adjusted after each classifier has been learned by the base learner (Dietterich, 2000). This process is demonstrated in **Figure 3-10**.

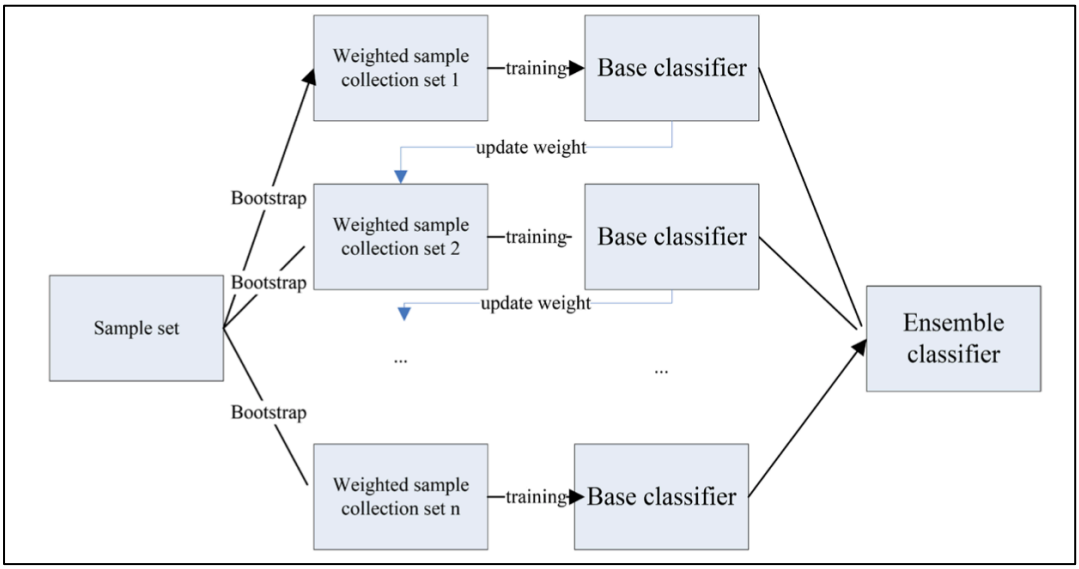

Figure 3-8. AdaBoost Algorithm Process Example (Li & Li, 2020)

For this predictive modeling application, AB parameters in **Table 3-4** indicate the base learner chosen as the "weak learner" to be improved (*DecisionTreeClassifier[max_depth=1], DecisionTreeClassifier[max_depth=2]*, set at two different settings to test various parameters), the learning rate or the weight assigned to each classifier at each boosting iteration (*1, 1.5, 2*; default is 1, but set others to test

various parameters), the number of estimators/trees in the forest at which boosting is terminated (*100, 200*; set to start at 100 and stop at 200), the selected boosting algorithm ("*SAMME,*" "*SAMME.R;*" the former is a "discrete" boosting algorithm and the latter is a "real" boosting algorithm which usually converges faster than the alternative and therefore produces a lower test error), and the random state (*3*, pseudo-random number set to achieve reproducible results on each run of the classifiers).

3.6.6 Gaussian Naïve Bayes for Predictive Modeling of Ventilator Recall Status

Regarded as one of the simpler, yet well-performing classifiers for continuous data, the NB algorithm outperforms more robust machine learning algorithms in a variety of applications (Raschka, 2014). In the context of this application, NB bases its class predictions on the assumption that the topic distributions (x) of a record (i), (x_i), follow a normal distribution. The probability of x_i belonging to a class c can be computed by the mathematical notation shown in (2) using the mean (μ) and variance (σ^2) of x_i:

$$P(x_i|c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp\left(-\frac{(x_i-\mu_y)^2}{2\sigma_c^2}\right) \quad (2)$$

For this application of NB, no parameters were specified, as the Scikit-learn library only allows for the specification of prior probabilities of classes and the variance smoothing parameter used to stabilize calculations. Pedregosa et al. (2011) noted that pre-specifying these parameters will not be justified in accordance with the dataset they are being defined for.

3.6.7 Logistic Regression for Predictive Modeling of Ventilator Recall Status

Similar to NB, the LR text classification algorithm is known for its simplicity, but is still considered an adequately performing classifier (Pranckevičius & Marcinkevičius, 2017). This classification method is considered to be a discriminative model, as it calculates the probability of belonging to a specific class, C_i, by discriminating among the various potential probabilities given an input vector, x_i and weight, w_i (Indra, Wikarsa, & Turang, 2016). Equation (3) represents the formula used by this classifier to compute the probability of a record belonging to class C_i, given an input vector, x_i, and N number of features.

$$P(c|x) = \frac{\exp\left(\sum_{i=1}^{N} w_i \cdot f_i(c,x)\right)}{\sum_{c' \in C} \exp\left(\sum_{i=1}^{N} w_i \cdot f_{i(c',x)}\right)} \quad (3)$$

For this classification algorithm, no parameters were specified, as the initial parameter option, described by Scikit-learn, is the regularization parameter, which dictates which other parameters can be used (Pedregosa et al., 2011). In the case of this praxis, the default "l2," or ridge, regularization option was preferred, because it performs best when there is no desire to eliminate features in the regularization process (El-Koka & Kang, 2016). This determination ultimately set all other parameter settings for this algorithm to default configurations.

3.6.8 Performance Evaluation of Models Used for Recall Status Prediction

After each model completed the training and testing procedures, their resulting macro F-1 scores were tabulated and the optimal parameters selected automatically by the code were summarized. The macro F-1 metric is regarded as one of the preferred

methods for classification model performance comparisons (Japkowicz, 2006; Toman, Tesar, & Jezek, 2006). The highest-scoring model using this metric was automatically chosen as the best-performing classifier and additional metrics were explored for its performance on the validation data and testing data separately as an additional check for overfitting. The model also allows for the manual input of other models to be selected as the "chosen model" for subsequent performance evaluations. The chosen model's precision, recall, F-1 score, and accuracy are regarded as the output of the RECALL_STATUS prediction phase of this praxis.

3.7 Predictive Modeling of Recall Classification Level from Ventilator Adverse Event Narratives

The third and final phase of this praxis is classification of the adverse event narrative reports of ventilators predicted to be recalled into their associated RECALL_STATUS_CLASSIFICATION category. In the end-to-end process illustrated in **Figure 3-1**, this approach is included in the *Predictive Modeling of Recalled Classification From Text Narratives* and *Model Performance* sections. The incoming arrow into these sections originates from the *Clean Data File* block. This phase was run using a Python code script, which can be found in **Appendix F**. The code was broken up by sections, with detailed commentary to facilitate reproducibility (statements starting with "#" explain the code's functionality step by step).

To accomplish the classification task for this step, a smaller subset from the original raw dataset was randomly chosen to mitigate the impact of the data imbalance caused by uneven RECALL_STATUS_CLASSIFICATION label distributions. As the

scope of this praxis is limited to ventilators, the amount of available data for this type of device served as a restriction. Once the subset was selected, records labeled as "Not Recalled" were deleted from the dataset, and the remaining records underwent the same preprocessing step referenced in **Section 3.3.1**. The final preprocessed subset was coded in a separate notebook titled "clean_data_1.pickle" and saved in a prespecified output folder. By saving the file in this format, it was easy to locate for subsequent use in other models and the file preserved its structure. The code utilized for this step is included in **Appendix C-6**.

3.7.1 Data Preparation for Predictive Modeling of Recall Classification Level

Following the preprocessing phase, the dataset was prepared for text classification. Classification labels were then defined for each potential recall classification option. **Table 3-5** details all labels in this subset of the data. For modeling purposes, each label was converted into a number (0, 1, 2) for classification.

Label Number	Recalled Status	Recall Class
0	RECALLED	1
1	RECALLED	2
2	RECALLED	3

Table 3-5. Defined Labels for Recall Classification Levels

The dataset then underwent a stratified sampling process to ensure that the same distribution of class labels was maintained for the training and testing sets. Following this step, 70% of the records were set aside for training and 30% for testing purposes. Akarsh,

Simran, Poornachandran, Menon, and Soman (2019) found that this split ratio of 70:30 provided better results when dealing with imbalanced datasets for machine learning applications when compared with a 80:20 ratio. The label proportions were again checked in the training dataset, and the label containing the second smallest number of records was used as the baseline sample size for all labels with a higher number of records.

Prior to loading relevant features into the classification models, the words within the adverse event narrative reports were vectorized to transform them into a format understandable by the algorithms. The resulting vectors were then scored using a tf–idf approach, which measures word relevance in one document within a collection of documents (Aizawa, 2003). Term frequency is first calculated by counting the number of times a word appears in a record, while inverse document frequency is calculated by counting the number of times a word appears in the collection of records. The tf-idf measure for a word in a record can then be calculated by multiplying these numbers together (Mishra & Vishwakarma, 2015). Common words that appear in many documents will have a tf-idf value close to 0, while rare words will have a tf-idf close to 1.

3.7.2 Classification Algorithms and Associated Parameters for Predictive Modeling of Recall Classification Levels for Ventilators

Similar to the text classification algorithms and accompanying parameter specifications listed in **Table 3-4**, the parameters specified for this predictive modeling application, shown again in **Table 3-6,** include most of the available options, thereby

allowing Python to perform a cross-validation grid search. By doing so, Python is able to find the best possible combination of parameter values for each algorithm (Pedregosa et al., 2011). To further combat the effects of imbalanced learning, a stratified k-fold cross-validation approach, with k=10 folds, was adopted to train the dataset. This value for k was suggested by Ma and He (2013) in their book detailing how to perform machine learning applications on imbalanced datasets.

Text Classification Algorithm	**Defined Parameters**
Stochastic Gradient Descent	loss = [‘log’,'hinge'], penalty = ['l2'], alpha = [1e-6, 1e-3, 1e-1, 1e0], max_iter = [5, 1000, 10000], tol = [None, 1e-3], random_state = [2])
Random Forest	bootstrap = [True, False], max_depth = [10, 50, 100, None], max_features = ['auto'], min_samples_leaf = [1, 2, 4], n_estimators = [500, 1000], random_state = [2])
AdaBoost	base_estimator = [DecisionTreeClassifier(max_depth=1), DecisionTreeClassifier(max_depth=2)], learning_rate = [1, 1.5, 2], n_estimators = [100, 200], algorithm = ["SAMME", "SAMME.R"], random_state = [2])

Gaussian Naïve Bayes	N/A—Default parameters used
Logistic Regression	N/A—Default parameters used (L2 Regularization)

Table 3-6. Text Classification Algorithms and Associated Parameters for Predictive Modeling of Recall Classification Levels Using Text Narratives

The end goal of this final phase of the praxis was to select the model that best performs when predicting the associated RECALL_STATUS_CLASSIFICATION (Class I, Class II, or Class III) from the text contained in the adverse event narrative reports for ventilators that have been predicted to be recalled.

3.7.3 Stochastic Gradient Descent for Predictive Modeling of Recall Classification Levels

The SGD algorithm's functionality remains the same as explained in **Section 3.6.3**. For this predictive modeling application, the SGD parameters were set as shown in **Table 3-6**. The SGD parameters in **Table 3-6** indicate the type of loss function used (*log*, gives a LR classifier with SGD training, and *hinge*, signifying the default parameter of a SVD classifier with SGD training), the penalty or regularization term to be used (set only at *l2*, the default parameter for SVD classifiers, as other settings may introduce feature vector sparsity to the model), alpha (*1e-6, 1e-3, 1e-1, 1e0*; constant value that multiplies the regularization term), tolerance (*1e-3*, the default stopping criterion, or *none*), and the random state (*2*, pseudo-random number set to achieve reproducible results on each run of the classifiers).

3.7.4 Random Forest for Predictive Modeling of Recall Classification Levels

The RF algorithm's functionality remains the same as explained in **Section 3.6.4**. For this predictive modeling application, the RF parameters were set as shown in **Table 3-6**. In this phase, the RF parameters in **Table 3-6** indicate the maximum depth/levels of each tree generated (*10, 50, 100, None*; set at various settings to test various parameters), the minimum number of samples required before splitting an internal node (*1, 2, 4*; default is 2 but included others to test various parameters), the number of estimators/trees in the forest (*500; 1,000*; set to start at 500 and stop at 1,000), and the random state (*2*, pseudo-random number set to achieve reproducible results on each run of the classifiers).

3.7.5 AdaBoost for Predictive Modeling of Recall Classification Levels

The AB algorithm's functionality remains the same as explained in **Section 3.6.5**. For this predictive modeling application, the AB parameters were set as shown in **Table 3-6**. The parameter settings included the base learner chosen as the "weak learner" to be improved (*DecisionTreeClassifier[max_depth=1], DecisionTreeClassifier [max_depth=2]*, set at two different settings to est various parameters), the learning rate or the weight assigned to each classifier at each boosting iteration (*1, 1.5, 2*; default is 1 but set others to test various parameters), the number of estimators/trees in the forest at which boosting is terminated (*100, 200*; set to start at 100 and stop at 200), the selected boosting algorithm ("*SAMME,*" "*SAMME.R;*" the former is a "discrete" boosting algorithm and the latter is a "real" boosting algorithm, which usually converges faster than the alternative and therefore produces a lower test error), and the random state (*2*, pseudo-random number set to achieve reproducible results on each run of the classifiers).

3.7.6 Gaussian Naïve Bayes for Predictive Modeling of Recall Classification Levels

Operating in the same manner as described in **Section 3.5.6**, NB was used in this context to classify the newly defined feature vectors into the various recall status classifications. No parameters were specified for this algorithm in this context, as the Scikit-learn library denotes default settings for the model to perform in accordance with the dataset being used.

3.7.7 Logistic Regression for Predictive Modeling of Recall Classification Levels

With the same base functionality as detailed in **Section 3.6.7**, the LR algorithm was used to classify the subset of ventilators predicted to be recalled into the various RECALL_STATUS_CLASSIFICATION labels using the textual data in the adverse event narrative reports and text length as feature vectors. Once again, the parameters were left at default settings and not specified in the code, as the regularization parameter dictated all other settings. L2 regularization, as well as its accompanying settings, was preferred in this praxis to eliminate the possibility of feature vectors being eliminated from the model.

3.7.8 Performance Evaluation of Models Used for Prediction of Recall Classification Level

After each model completed the training and testing procedures on the dataset, their resulting macro-F1 scores were tabulated and a summary of the parameters deemed "optimal" by the Python code was displayed. Overfitting was also verified by an evaluation of each model's precision, recall, F1 score, and accuracy on the training and

testing data separately. The highest-scoring model using the macro-F1 score and accuracy was automatically chosen as the best-performing classifier. The Python code also allowed for the manual input of other models to be selected as the "chosen model" for subsequent performance evaluations. Finally, the chosen model's precision, recall, F1 score and accuracy are regarded as the output of the RECALL_STATUS_CLASSIFICATION prediction portion of this praxis.

3.8 Summary

This chapter summarized the data collection process, including FDA file linking, and the cleaning and preprocessing steps required to prepare the dataset for analysis. The LDA topic modeling approach was discussed for the detection of topics associated with ventilator malfunctions from FDA MAUDE adverse event narrative reports. After determination that the majority of topics identified signify early warning signs of ventilator defects, this chapter explained how the topics could be utilized to uncover useful recall status information for manufacturers. The topics' ability to serve as feature vectors in a series of machine learning classifiers was explored for the automated prediction of recalls. Finally, the process of applying machine learning predictive modeling to adverse event narratives was employed in a novel approach to determine recall classification levels for recalled ventilator devices.

Chapter 4—Results

4.1 Introduction

This chapter presents the results obtained from the three phases of this praxis: Latent Dirichlet Allocation (LDA) topic model for early warning sign detection of ventilator defects, predictive modeling for ventilator recall status from identified early warning signs of defects, and predictive modeling for applicable recall classification level from ventilator adverse event narratives. The results gathered from the aforementioned machine learning applications area should assist ventilator manufacturers in the process of detecting early warning signs of ventilator defects and thereby mitigate the potential negative consequences of device recalls.

The first model involves the use of the LDA topic model to extract the early warning signs of ventilator defects from the adverse event text narrative reports submitted by end users of this type of medical device. A randomly selected sample of ventilator device brands was explored, and the resulting extracted topics identified by the LDA model were presented. Their relevance to early warning signs of ventilator defects was evaluated by a panel of subject matter experts (SMEs) with varying levels of expertise in the fields of machine learning and medical device design. Following assertion of the validity of the identified topics as early warning signs, the next model adopted the topic distributions of each record in the processed dataset from the LDA model as feature vectors to categorize each report by the associated recall status of the device. The final model predicted the recall classification level of a randomly selected subset of records

that were predicted to be recalled. Both predictive modeling approaches underwent a stratified k-fold cross-validation process to further validate the obtained results.

4.2 Phase One: Latent Dirichlet Allocation Topic Modeling for Early Warning Sign Detection of Ventilator Defects

Latent Dirichlet Allocation topic modeling was used in this study to determine which words or phrases are most salient in the adverse event narrative reports submitted by end users. These topics were extracted by the model and their connection to ventilator defects was assessed by SMEs to determine whether they are indicative of early warning signs of defects. The devices analyzed by the LDA model included a mixture of life-sustaining and non-life-sustaining ventilators, as well as models that are classified within all of the risk category classifications defined by the FDA. The LDA model generated a list of 14 topics with 10 words each. A team of machine learning and medical device design experts reviewed the full list to determine if the words identified were relevant to ventilator devices and associated defects. In line with the methodology proposed by Fong, Sarkani, and Fossaceca (2021), in which SMEs validated the LDA model results, in this praxis, the final determination of the SMEs was used to validate the LDA model's results and test the hypothesis formulated for this step.

4.2.1 Optimal Amount of Topics Selected

A coherence score was utilized in this study to determine the optimal number of topics to consider from the LDA model. The coherence model in Python generated a plot showcasing the coherence score against the number of topics evaluated by the LDA model. As suggested by Islam and Goldwasser (2020) in their study on LDA topic

modeling, the optimal amount of topics lies at the end of a period of rapid growth in topic coherence. From **Figure 4-1** and **Table 4-1**, it is evident that there are two distinct points where this occurs, when the number of topics is 4 and 14.

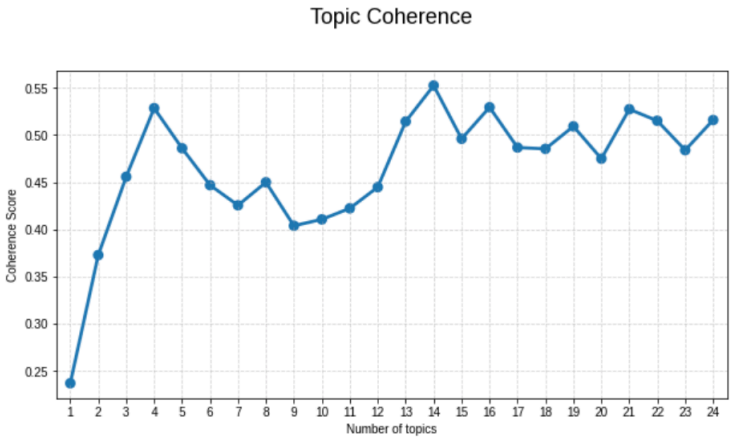

Figure 4-1. Topic Coherence Graph

Number of Topics	Coherence Scores
1	0.237
2	0.373
3	0.456
4	0.529
5	0.486
6	0.447
7	0.426

8	0.450
9	0.404
10	0.411
11	0.422
12	0.445
13	0.514
14	0.553
15	0.496
16	0.530
17	0.487
18	0.485
19	0.509
20	0.475
21	0.527
22	0.515
23	0.484
24	0.516

Table 4-1. Coherence Scores by Number of Topics

To further explore these two options, they were evaluated using a pyLDAvis interactive visualization. The intertopic distance maps in **Figure 4-2** demonstrate how topic distributions interact with each other in a 2D space. Large bubbles indicate a high prevalence of a specific topic in the dataset, but may prevent users from distinguishing between predominant themes in the dataset (Argyrou, Giannoulakis, & Tsapatsoulis, 2018). Overlapping bubbles indicate shared themes by topics. The distance map on the left references a topic model with only four topics, so the bubbles are large and mostly spaced out, which indicates that the model has grouped a large amount of words into each of the topic distributions (Islam & Goldwasser, 2020). On the right side, however, there are considerably more bubbles present, which are spaced out throughout the quadrants on the distance map, and the overlap is still minimal. This second map provides less-generalized topic distributions, as there are a larger amount of individual topics.

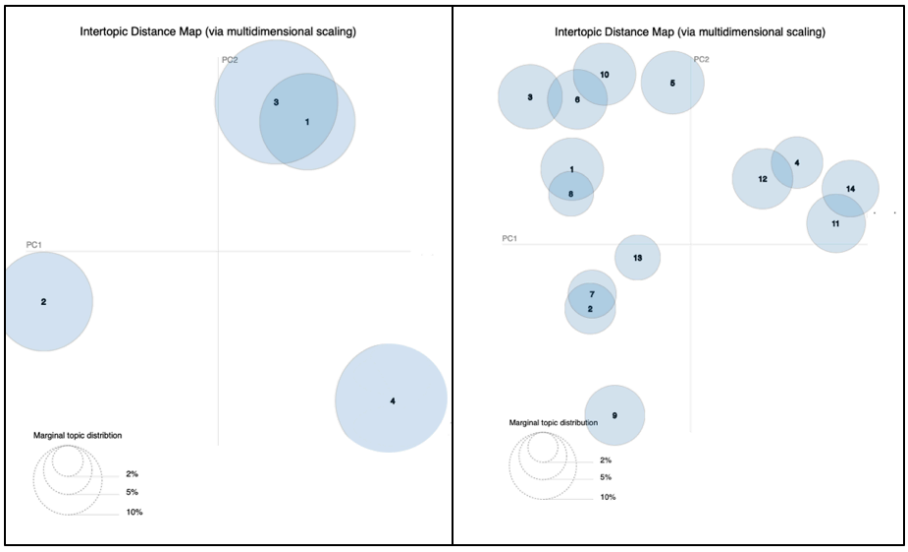

Figure 4-2. Intertopic Distance Maps with Number of Topics at 4 (left) and with Number of Topics at 14 (right)

As choosing optimal topic amounts is a trial-and-error process, the suggestion of most practitioners is to utilize the information obtained from the topic coherence plot as well as the intertopic distance map to make the final selection (Stevens, Kegelmeyer, Andrzejewski, & Buttler, 2012). Therefore, 14 topics was selected as the optimum number of topics for the LDA model, given the highest coherence score of 0.553 with fairly large and minimally overlapping topic bubbles that do not cluster in only one quadrant of the graph.

4.2.2 Topics Generated by the Latent Dirichlet Allocation Model

The LDA topic model was used to identify the 14 most prevalent topics in the adverse event narrative reports submitted by end users on a mixture of ventilator devices. The top 10 most salient words or word phrases identified per topic are listed in

Table 4-2. The support weight for each is also included, indicating the representativeness of a word in its assigned topic (Lee, Kim, & Myaeng, 2015). The full list of topics and supporting words was provided to a team of SMEs in the fields of machine learning and medical device design.

The team was asked to identify which of the words were relevant to ventilator devices or associated with device defects. As the diverse knowledge base and experiences of the SMEs contributed to different perspectives on what should be considered an early warning of ventilator defects, a group discussion was conducted to discuss the LDA model's results. During this discussion, the SMEs provided their feedback on each extracted term and collectively came to a final conclusion as to which words were considered early warning signs for ventilator defects. Per their feedback, the factors

considered for their assessment included the following:

- words that indicate conditions where a defect might be present in a device component,
- words associated with software terms or software defects,
- words that have high probabilities of being considered descriptors of a defect,
- words typically used to describe root-cause investigations of malfunctions,
- words referring to end-user experiences.

The words deemed to be relevant to early warning signs of ventilator defects are highlighted in grey in **Table 4-2**. This list was used to validate the results obtained by the LDA topic model and reinforce the usefulness of the latent topics as early warning signs of ventilator defects in additional machine learning applications explored in this praxis.

The last column in **Table 4-2** provides a total count of words that were considered indicative of ventilator defects per topic. The list of words identified by the team of experts as early warning signs has a mean of 7.57 out of 10 words and a median of 8 out of 10 words evaluated as providing indication of early warning signs of ventilator defects. These results were utilized to test Hypothesis I, as discussed in **Section 5.2.1**.

Topics	Words and Support Weights		Words and Support Weights		Words and Support Weights		Total Relevant to Early Warning Signs of Ventilator Defects
1	engine	0.034	occlusion	0.019	transducer	0.018	8 of 10
	mode	0.012	corrective	0.012	required	0.012	
	gas_delivery_engine	0.011	remediation	0.011	field_corrective_action	0.010	
	center	0.009					
2	diaphragm	0.026	_service_engineer	0.025	cap	0.019	8 of 10
	exhalation_flow_sensor	0.019	memory	0.018	evq	0.016	
	relevant	0.015	diagnostic_code	0.014	inoperable	0.013	
	volume	0.012					
3	button	0.016	meter	0.016	user_interface_module	0.014	9 of 10
	touch	0.014	touchscreen	0.013	panel	0.010	
	carefusion_failure_analysis_lab	0.010	control	0.009	start_/_stop_button	0.009	
	suspect_component	0.008					
4	cpap	0.047	system	0.032	bubble	0.031	8 of 10
	probe	0.021	air	0.018	distributor	0.016	
	generator	0.016	kit	0.011	foreign_distributor	0.011	
	board	0.008					
5	user_facility	0.033	info	0.028	specialist	0.019	4 of 10
	tech	0.019	document	0.019	distributor	0.015	
	carefusion_tech_support_specialist	0.013	eval	0.012	phone	0.012	
	base	0.011					
6	blender	0.019	carefusion_field_service_representative	0.015	mean	0.010	9 of 10
	onsite	0.010	mean_airway_pressure	0.010	map	0.009	
	device_onsite	0.009	switch	0.009	setting	0.009	
	manufacturer_specification	0.009					
7	serial	0.029	death	0.020	usb	0.017	8 of 10
	transfer	0.015	graphic	0.015	bus	0.013	
	line	0.013	software	0.012	_service_engineer	0.010	
	flash	0.010					
8	water	0.019	turn	0.016	occlusion	0.019	10 of 10
	switch	0.012	smell	0.010	capacitor	0.008	
	transducer	0.008	shut	0.008	audible	0.007	
	cable	0.007					
9	breath	0.029	failure_investigation	0.024	returned_component	0.024	8 of 10
	summary	0.022	test_ventilator	0.022	errors	0.019	
	device_evaluation_summary	0.018	manufacturing_specification	0.018	anomalies	0.017	
	controller	0.016					
10	call	0.021	machine	0.020	tech	0.019	6 of 10
	specialist	0.017	document	0.015	medwatch_report	0.014	
	phone	0.014	carefusion_tech_support_specialist	0.014	knob	0.013	
	conversation	0.013					
11	tubing	0.031	engineering	0.021	fire	0.018	8 of 10
	risk	0.016	damage	0.015	engineering_investigation	0.015	
	catch	0.013	returned	0.011	alleged_malfunction	0.010	
	require	0.010					
12	bag	0.030	sample	0.016	allege	0.012	8 of 10
	defect	0.009	headgear	0.009	resuscitator	0.007	
	bonnet	0.007	infant	0.007	room	0.007	
	staff	0.007					
13	reportable	0.035	medwatch_report	0.034	customer_complaint	0.026	4 of 10
	based	0.021	initial	0.020	previously	0.019	
	regulation	0.018	release	0.018	guidance	0.017	
	deem	0.017					
14	nasal	0.063	infant	0.049	tubing	0.031	8 of 10
	prongs	0.026	prong	0.018	paykel_healthcare	0.016	
	cpap	0.011	nasal_tubing	0.009	damage	0.009	
	instruction	0.008					

Table 4-2. Topics Identified, Salient Words, Support Weights, and Relevance to Early Warning Signs of Ventilator Defects

The final list of early warning signs of ventilator defects, as validated by the SMEs, is presented in **Table 4-3.**

engine	corrective	field_corrective_action	button	system	start_/_stop_button	breath	release	allege
gas_delivery_engine	remediation	cap	carefusion_failure_analysis_lab	air	bubble	device_evaluation_summary	prong	resuscitator
center	memory	evq	suspect_component	kit	mean	controller	nasal_tubing	room
diaphragm	meter	inoperable	probe	eval	setting	call	regulation	damage
exhalation_flow_sensor	touchscreen	user_interface_module	generator	carefusion_field_service_representative	usb	tubing	nasal	cpap
volume	control	panel	board	mean_airway_pressure	bus	risk	prongs	instruction
base	death	_service_engineer	line	manufacturing_specification	anomalies	catch	reportable	bag
blender	software	occlusion	flash	machine	tech	bonnet	staff	defect
manufacturer_specification	shut	returned_component	transducer	headgear	engineering_investigation	onsite	turn	capacitor
serial	failure_investigation	errors	cable	medwatch_report	alleged_malfunction	device_onsite	smell	audible
water	carefusion_tech_support_specialist	knob	switch	returned	fire			

Table 4-3. Early Warning Signs of Ventilator Defects

4.3 Phase Two: Predictive Model for Ventilator Recall Status Using Early Warning Signs of Ventilator Defects

After the extracted topics from the LDA model and, consequently, the trained LDA model were validated as indicative of early warning signs of ventilator defects, these could be utilized within other machine learning applications. The second phase of this praxis involved the use of predictive modeling to determine whether a record in the dataset would be classified as “recalled” or “not recalled” using the identified early

warning signs as feature vectors for the models. As described in **Section 3.6**, this process was initiated by testing out various machine learning algorithms to determine which performs best under varying model parameters for this particular application. The records classified as "recalled" are those that are labeled as "Y" in the RECALL_STATUS column and "not recalled" records are those that are labeled as "N" in the RECALL_STATUS column. The final dataset used for this phase included 2,399 records labeled as "Y" and 1,593 records labeled as "N." The data were subject to a stratified 15-fold cross-validation process; thus, the models were trained with 3,193 records, with 639 of these used for validation. The records utilized for testing amounted to 20% of the dataset, or 799 records. Those 799 remained unseen by the models until model training had been completed. The training and testing model accuracy over varying levels of folds, as well as corresponding processing times, are shown in **Table 4-4**.

K-Folds Cross Validation	Processing Time (in sec.)	Models	Training Accuracy	Testing Accuracy
5-fold	1,348	Random Forest	0.793	0.806
		AdaBoost	0.786	0.798
		Logistic Regression	0.778	0.806
		Stochastic Gradient Descent	0.776	0.807
		Gaussian Naïve Bayes	0.756	0.778
10-fold	2,886	Random Forest	0.793	0.806
		AdaBoost	0.784	0.797
		Logistic Regression	0.778	0.806
		Stochastic Gradient Descent	0.778	0.809
		Gaussian Naïve Bayes	0.756	0.778
15-fold	4,504	Random Forest	0.798	0.820

		AdaBoost	0.784	0.797
		Logistic Regression	0.778	0.806
		Stochastic Gradient Descent	0.778	0.809
		Gaussian Naïve Bayes	0.756	0.778
20-fold	5,960	Random Forest	0.797	0.807
		AdaBoost	0.786	0.798
		Logistic Regression	0.778	0.806
		Stochastic Gradient Descent	0.778	0.809
		Gaussian Naïve Bayes	0.756	0.778

Table 4-4. Processing Times and Training/Validation and Testing Accuracy Over Varying Levels of Folds for Classification of Recall Status

With the highest-achieved performances by the random forest classifier in both validation and testing of 79.8% and 82.0%, respectively, a value of 15 folds was selected for this specific phase of the praxis. The results obtained for this classifier operating under this selected value of folds are shaded in grey on **Table 4-4**.

4.3.1 Model Performance for Ventilator Recall Status Classification

The Python code designed for this phase of the praxis, provided in **Appendix E**, evaluates algorithms based on their predictive ability to classify records as pertaining to "recalled" or "not recalled" ventilator devices by attempting various model parameters as specified prior to code execution. After a detailed evaluation of the models' macro-F1 scores and accuracy, the code makes the decision as to which model performed best overall. The calculations for each of these metrics, as well as precision and recall, were completed following the confusion matrix presented in **Figure 4-3**.

		Predicted Values	
		Recalled	*Not Recalled*
Actual Values	*Recalled*	True Positives	False Negatives
	Not Recalled	False Positives	True Negatives

Figure 4-3. 2x2 Confusion Matrix Used for Classification of Recall Status

Table 4-5 highlights the performance results on training/validation data accuracy over the specified 15 folds and testing data accuracy for all models executed.

Models	Training/Validation Accuracy	Testing Accuracy
Random Forest	0.798	0.820
AdaBoost	0.784	0.797
Logistic Regression	0.778	0.809
Stochastic Gradient Descent	0.778	0.806
Gaussian Naïve Bayes	0.756	0.778

Table 4-5. Training/Validation Data Accuracy Results After 15-Fold Cross Validation for Classification of Recall Status

Table 4-6 provides detailed performance results for the models' ability to predict "recalled" and "not recalled" devices from the training data using the remaining performance metrics. Finally, **Table 4-7** provides the remaining performance metrics results based on the models' ability to predict "recalled" and "not recalled" devices from the testing data.

Models	RECALL_STATUS	Precision	Recall	F1-score	Accuracy
Random Forest	Recalled	0.831	0.833	0.832	
	Not Recalled	0.748	0.745	0.747	0.798
AdaBoost	Recalled	0.829	0.807	0.818	
	Not Recalled	0.721	0.749	0.735	0.784
Logistic Regression	Recalled	0.829	0.794	0.811	
	Not Recalled	0.708	0.753	0.730	0.778
Stochastic Gradient Descent	Recalled	0.831	0.792	0.811	
	Not Recalled	0.707	0.757	0.731	0.778
Gaussian Naïve Bayes	Recalled	0.865	0.703	0.776	
	Not Recalled	0.651	0.835	0.732	0.756

Table 4-6. Additional Performance Metrics Results for Training/Validation Data After 15-Fold Cross Validation for Classification of Recall Status

Models	RECALL_STATUS	Precision	Recall	F1-score	Accuracy
Random Forest	Recalled	0.840	0.865	0.852	
	Not Recalled	0.787	0.752	0.769	0.820
AdaBoost	Recalled	0.826	0.840	0.833	
	Not Recalled	0.752	0.734	0.743	0.797
Logistic Regression	Recalled	0.838	0.840	0.839	
	Not Recalled	0.758	0.755	0.757	0.806
Stochastic Gradient Descent	Recalled	0.841	0.840	0.840	
	Not Recalled	0.759	0.762	0.761	0.809
Gaussian Naïve Bayes	Recalled	0.860	0.754	0.804	
	Not Recalled	0.688	0.815	0.746	0.778

Table 4-7. Additional Performance Metrics Results for Testing Data for Classification of Recall Status

The results obtained for the highest-performing model for this specific phase of the praxis

are summarized in **Table 4-8** and were used to test Hypothesis II, as discussed in **Section 5.2.2**. These results were achieved under the following model parameters: bootstrap = [True], max_depth = [50], max_features = [“auto”], min_samples_leaf = [2], n_estimators = [500].

Highest Performing Model	RECALL_STATUS	Precision	Recall	F1-score	Accuracy
Random Forest	Recalled	0.840	0.865	0.852	0.820
	Not Recalled	0.787	0.752	0.769	

Table 4-8. Summary of Highest-Performing Model Metrics for Classification of Recall Status

4.4 Phase Three: Classification Model for Class Prediction from Adverse Event Narratives

The final phase of this praxis explored the use of predictive modeling algorithms, but using the actual textual narratives of adverse event reports that have already been classified as “recalled” as feature vectors for classification models. The goal of this step was to predict which records would fall within each recall classification status (Class I, Class II, or Class III). **Section 3.7** explains how this process was executed by testing various machine learning algorithms to determine which performs best under varying model parameters. The model performances were evaluated, and the best-performing model for this specific application was selected. As the documented recall classification levels of ventilator devices were imbalanced, for this additional step, a randomly selected subset of the original dataset was used. Following the guidance of Akarsh, Simran,

Poornachandran, Menon, and Soman (2019) for training/testing ratio selections for imbalanced learning, a 70:30 ratio was adopted for this phase of the praxis with a total of 616 records for model training/validation and 264 records for model testing. The testing set remained unseen by all models until model training and validation had been completed. In addition, as the general recommendation for imbalanced datasets is to employ a stratified 10-fold cross-validation strategy during training, this strategy was implemented for this phase of the praxis (Japkowicz, Ma & He, 2013). Japkowicz, Ma, and He (2013) noted how the use of this approach in imbalanced datasets guarantees that the ratio of positive to negative records is preserved in all 10 folds. Furthermore, because the subset of data used contained fewer records than the original dataset, processing time was a negligible factor for this phase.

4.4.1 Model Performance for Ventilator Recall Classification Level

In this phase, the Python code evaluated algorithms based on their predictive ability to classify "recalled" records according to their applicable recall classification level (Class I, Class II, or Class III). This evaluation was performed by attempting various model parameters specified prior to code execution, as explained in **Section 3.7.2**, and allowing Python to perform a grid search for all possible parameter combinations. After a detailed evaluation of the models' macro-F1 scores and accuracy, the code makes the decision as to which model performed best overall. In this case, as there are three classification levels, a 3x3 confusion matrix was used to evaluate model performance, as illustrated in **Figure 4-4**. The values represented in this confusion matrix are as follows: A=Class I, B=Class II, C=Class III, TAA=True Class I, TBB=True Class II, TCC=True Class III, FAB=False Class I/True Class II, FAC=False Class I/True Class

III, FBA=False Class II/True Class I, FBC=False Class II/True Class III, FCA=False Class III/True Class I, and FCB=False Class III/True Class II.

		Predicted Values		
		A	*B*	*C*
Actual Values	*A*	TAA	FBA	FCA
	B	FAB	TBB	FCB
	C	FAC	FBC	TCC

Figure 4-4. 3x3 Confusion Matrix Used

Table 4-9 highlights the performance results on training/validation data accuracy over the specified 10 folds and testing data accuracy for all models executed.

Models	**Training/Validation Accuracy**	**Testing Accuracy**
Stochastic Gradient Descent	0.915	0.894
Random Forest	0.880	0.905
Logistic Regression	0.859	0.898
AdaBoost	0.838	0.864
Gaussian Naïve Bayes	0.859	0.852

Table 4-9. Training/Validation Data Accuracy After 10-Fold Cross Validation and Testing Data Results for Prediction of Recall Status Classification Level

Table 4-10 provides detailed performance results for the models' ability to predict the associated recall classification level from the training data using the remaining

performance metrics.

Models	RECALL_STATUS_ CLASSIFICATION	Precision	Recall	F1-score	Accuracy
Stochastic Gradient Descent	Class I	0.864	0.966	0.912	
	Class II	0.980	0.847	0.909	
	Class III	0.920	0.958	0.939	0.915
Random Forest	Class I	0.828	0.898	0.862	
	Class II	0.914	0.898	0.906	
	Class III	0.950	0.792	0.864	0.880
Logistic Regression	Class I	0.810	0.864	0.836	
	Class II	0.881	0.881	0.881	
	Class III	0.950	0.792	0.864	0.859
AdaBoost	Class I	0.794	0.847	0.820	
	Class II	0.862	0.847	0.855	
	Class III	0.905	0.792	0.844	0.838
Gaussian Naïve Bayes	Class I	0.812	0.881	0.846	
	Class II	0.893	0.847	0.870	
	Class III	0.909	0.833	0.870	0.859

Table 4-10. Additional Performance Metrics Results for Training/Validation Data After 10-Fold Cross Validation for Prediction of Recall Status Classification Level

Finally, **Table 4-11** provides the remaining performance metrics results based on the models' ability to predict the associated recall classification level from the testing data.

Models	RECALL_STATUS_ CLASSIFICATION	Precision	Recall	F1-score	Accuracy
Stochastic Gradient Descent	Class I	0.941	0.877	0.908	0.894
	Class II	0.826	0.905	0.864	
	Class III	0.889	0.941	0.914	
Random Forest	Class I	0.929	0.897	0.913	0.905
	Class II	0.878	0.940	0.908	
	Class III	0.879	0.853	0.866	
Logistic Regression	Class I	0.934	0.877	0.905	0.898
	Class II	0.835	0.964	0.895	
	Class III	0.933	0.824	0.875	
AdaBoost	Class I	0.883	0.877	0.880	0.864
	Class II	0.826	0.845	0.835	
	Class III	0.879	0.853	0.866	
Gaussian Naïve Bayes	Class I	0.897	0.836	0.865	0.852
	Class II	0.768	0.869	0.816	
	Class III	0.909	0.882	0.896	

Table 4-11. Additional Performance Metrics Results for Testing Data for Prediction of Recall Status Classification Level

The results obtained for the highest-performing model for this specific phase of the praxis are summarized in **Table 4-12** and were used to test Hypothesis III, as discussed in **Section 5.2.3**. These results were achieved under the following model parameters: loss = ["hinge"], penalty = ["l2"], alpha = [1e-3], max_iter = [1,000], tol = [None].

Highest Performing Model	RECALL_STATUS_ CLASSIFICATION	Precision	Recall	F1-score	Accuracy
Stochastic Gradient Descent	Class I	0.941	0.877	0.908	0.894
	Class II	0.826	0.905	0.864	
	Class III	0.889	0.941	0.914	

Table 4-12. Summary of Highest-Performing Model Metrics for Prediction of Recall Status Classification Level

4.5 Summary of Results

This chapter presented all results obtained from this research. The results from the LDA topic modeling phase indicated that the optimal number of topics was 14, as this amount provided the highest coherence score within the topics and relevant words extracted. The median number of relevant words identified as early warning signs of ventilator defects was 8 out of 10, further supporting the notion that these latent topics are indicative of ventilator defects. A total of 96 words were validated by SMEs as early warning signs of ventilator defects. For the second phase of this praxis, the random forest classifier was observed to be the highest-performing model for the prediction of recall status, achieving a macro F1-score of 0.811 and 82.0% accuracy. Finally, for the models employed to predict recall status classification levels, the SVM classifier trained with SGD was the highest-performing model, with a macro F1-score of 0.895 and an accuracy of 89.4%.

Chapter 5—Discussion and Conclusions

5.1 Discussion

The purpose of this research was to explore early warning signs of ventilator recalls through the use of various machine learning approaches. To accomplish this goal, three research questions were posed: Can Latent Dirichlet Allocation (LDA) topic modeling identify early warning signs of ventilator defects using adverse event narrative data? Can the identified early warning signs be used as feature vectors in machine learning models to predict ventilator recalls? Can machine learning utilize adverse event raw narrative data to predict all classes of recall types for ventilator devices? The results of this praxis indicated that all of the machine learning techniques executed were successful at accomplishing their intended purpose and that a sequential combination of all three yields useful results for manufacturers and product design experts.

The first research question addressed whether the use of topic modeling could be utilized to identify early warning signs of ventilator defects from the adverse event narrative reports submitted by end users to the U.S. Food & Drug Administration (FDA). The LDA model successfully extracted 14 topics of 10 words each, with a median score number of 8 out of 10 words suggesting an early warning sign of ventilator defects. Of the 140 total words included within these topics, 96 were validated by SMEs as early warning signs of ventilator defects. The median number of defect-related words was the measure employed to test the hypothesis associated with this first research question, following the baseline established by Fong, Sarkani, and Fossaceca (2021) in a similar study. The variety of topics uncovered by this model provided multiple sources of

defects. For example, Topic 3 was composed of 9 out of 10 words classified by experts as relevant to defects, among which were the words “button,” “touchscreen,” “control,” and “panel,” as well as the trigram “user_interface_module.” These words are indicative of a defect pertaining to faulty responsiveness of the device during its use; a quick word search for the term “interface” returned 472 records within the dataset, alluding to internal device communication failures. From the results obtained by the LDA model, a specific record in the dataset referenced an “AVEA Ventilator,” with Topic 3 contributing the highest attributable probability of belonging to the record. As per the Medical Device Recall database,[1] the “AVEA” ventilator was effectively recalled due to a potential risk of an incorrectly attached fuse that may cause “loss of power to the User Interface Module (UIM).”

The second research question sought to determine whether further use of the LDA model’s results could serve as predictive modeling feature vectors to mitigate potential product recalls. The models' performance results obtained using the distribution of topics assigned to each record provided higher levels of accuracy than those achieved by a prior study focused on medical data classification using LDA results (Nuser & Al-Horani, 2020). This combination of topic modeling and predictive modeling served to provide insights into potential product recalls, but its performance was hindered when trying to classify records by the various recall classification levels. This difficulty was assumed to occur because of two major factors: (1) the existing overlap within certain topics alluding to similar ventilator defects, which prevented the model from identifying which

[1] Class 1 Device Recall AVEA Standard with Compressor ventilator https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfRES/res.cfm?id=146311

distribution of topics belonged to each recall classification level, and (2) the data imbalance that exists within the three recall classification levels. To address these limitations, a third machine learning approach was explored using the raw textual data as feature vectors to predict recall classification levels, instead of topic distributions.

The third research question sought to answer whether recall classification levels could be predicted using the raw textual narrative data found in a subset of the adverse event reports. Overall, the Stochastic Gradient Descent (SGD) trained on a Support Vector Machine (SVM) classifier proved to be the highest-performing model for this approach, with a macro F1-score of 0.938 and an accuracy of 88.6%. In addition, the overall performance achieved by the various algorithms explored—with macro F1-scores ranging from 0.867 to 0.938 and accuracy results ranging from 85.2% to 90.5%—demonstrated that this approach is well-suited to predict recall classification levels. The algorithms explored for this purpose surpassed the performance levels set as a baseline goal. Thus, it was determined that manufacturers and medical device engineers can utilize this approach to gain a more detailed outlook on potential ventilator recalls.

5.2 Conclusions

Along with the three research questions presented at the beginning of this study, three accompanying hypotheses were formulated. Utilizing prior studies' results, the hypotheses were designed to test whether the methods explored could be improved when used in the context of medical device defect detection. The following subsections present the tests performed for each of the hypotheses and their accompanying statistics.

5.2.1 Hypothesis I

H1: Topic modeling will be able to identify 70% of early warning signs for ventilator defects using adverse event narrative data.

Hypothesis 1 evaluated the overall performance of the LDA topic model in its ability to successfully extract at least 70% of early warning signs of ventilator defects from the adverse event narrative reports submitted by end users. Since there were 14 topics extracted by the LDA topic model and evaluated by SMEs, the test carried out in this step was a one-sample sign test for the median number of words evaluated as indicative of potential early warning signs of ventilator defects. The baseline median was set at 70%, or 7 out of 10 words, in line with a similar scheme presented by Fong, Sarkani, and Fossaceca (2021) that demonstrated successful performance of LDA topic modeling when used to extract product defect information from narrative feedback data. The list of words identified by the team of experts as early warning signs had a mean of 7.57 out of 10 words and a median of 8 out of 10 words evaluated as providing indication of early warning signs of ventilator defects. The test statistic for this one-sample sign test was determined by selecting the minimum of two alternatives: the number of topics that included more than 7 relevant words or the number of topics that included less than 7 relevant words. As only 3 topics included less than 7 relevant words, the test statistic was determined to be 3. Using the binomial probability distribution table included in **Appendix G**, at a significance level of 0.05 with 14 topics, the critical region to reject the null hypothesis includes 3. Therefore, the one-sample sign test rejected the null hypothesis; the LDA topic model was successful at extracting early warning signs of ventilator defects. Furthermore, as the overall coherence score of the model was

determined to be 0.553, the topics, as well as the words associated with them, were further validated as being semantically coherent.

As these results surpassed the stipulated 70% baseline median, the uncovered topics, the words assigned to each, and the trained LDA model are deemed successful at extracting early warning signs of ventilator defects. As they were further validated by SMEs, these words, as well as the LDA model trained on them, can be effectively used for other applications regarding ventilator defect detection. These findings support the use of topic modeling approaches for defect detection within the medical device industry.

5.2.2 Hypothesis II

H2: The identified early warning signs can be used as feature vectors in classification models to predict potential recalls with 75% accuracy.

Hypothesis 2 evaluated the performance of the best-performing model with regards to its ability to classify "recalled" and "non-recalled" records, as determined by the Python code. The best-performing model for this application was determined to be the Random Forest (RF) classifier. The test data that remained unseen by the model were predicted after the 15-fold cross-validation process was completed and then compared to the actual classification label. The hypothesis test utilized was a one sample Z-test for a proportion at an alpha level of 0.05 with 75% accuracy. This target level of accuracy was determined by analyzing results from a comparable study that evaluated the ability of LDA topic modeling to be used in the context of medical text classification and achieved an accuracy level of 71% (Nuser & Al-Horani, 2020). At a significance level of 0.05, Hypothesis 2 was tested with regards to the best-performing model's ability to successfully predict "recalled" and "not recalled" records with an accuracy level of 75%.

With a p-value <0.0001, the hypothesis test successfully rejected the null hypothesis; therefore, this model achieved the expected result of identifying "recalled" ventilator devices from among those that are "not recalled."

From these findings, it can be deduced that the use of topic modeling in conjunction with predictive modeling approaches provides valuable insights into the product recall determination process. Using these two approaches, ventilator manufacturers can efficiently obtain useful information about their products' defects and mitigate the potential for recalls in a proactive manner.

5.2.3 Hypothesis III

H3: Machine learning can predict the associated recall classification level of defective ventilators with 85% accuracy.

Hypothesis 3 evaluated the performance of the best-performing model, as determined by the Python code, with regards to its ability to correctly predict which "recalled" records belong to each recall classification level. In this case, the best-performing model was determined to be the Support Vector Machine (SVM), with Stochastic Gradient Descent (SGD) training, classifier. The test data that remained unseen by the model were predicted after the 10-fold cross-validation process was completed and then compared to the actual classification label. The hypothesis test employed was a one sample Z-test for a proportion at an alpha level of 0.05 with 85% accuracy. This target level of accuracy was determined by analyzing results from a comparable study that used machine learning classifiers to determine the recall status of medical devices, which achieved a highest model accuracy of 84% (Emakhu, Aguwa,

Monplaisir, & Arslanturk, 2020). At a significance level of 0.05, Hypothesis 3 was tested with regards to the best-performing model's ability to successfully predict the associated recall classification level of recalled ventilator devices with an accuracy of 85%. With a p-value of 0.023, the hypothesis test successfully rejected the null hypothesis; therefore, this model achieved the expected result of classifying recalled ventilator devices by their associated classification level (Class I, Class II, or Class III).

This step is considered a novel approach to medical device recall classification level prediction. As the hypothesis test confirmed that the highest accuracy achieved by the models surpassed the minimum performance threshold, this approach proved to be an effective way of predicting the recall classification level for ventilator devices. This resource will allow manufacturers to efficiently mine through adverse event narrative reports of their products and gain useful information about potential recall events in a proactive manner.

5.3 Contributions to Body of Knowledge

This praxis explored a sequential combination of topic modeling and machine learning predictive models to reveal which early warning signs of ventilator defects could help identify a potential recall of these devices. Despite growing dependence on ventilators in response to the current Coronavirus Disease 2019 pandemic, the literature review was unable to provide any indication of similar work conducted in the medical device field prior to this study.

Furthermore, the additional step to classify the records that were predicted to be recalled into recall classification levels has never been explored by other researchers. In fact, the literature review only uncovered one research project that made determinations

based on a single recall classification level (Ensign & Cohen, 2017). In summary, this praxis adds to the body of knowledge in the medical device field by contributing a novel approach that combines a series of machine learning and topic modeling techniques to detect early warning signs of defects, anticipate the potential for recalls, and classify potential recalls based on risk level.

5.4 Recommendations for Future Research

Although the Coronavirus Disease 2019 pandemic was the main motivating factor for the research objectives explored in this praxis, the machine learning approaches discussed can be applied to uncover early warning signs of defects for any drugs or medical devices. Future research endeavors can adopt the proposed methodology while easily altering the test parameters prior to running the code to fit the specific needs of each study. In addition, as the methodology lends itself to be fine-tuned and applied easily within other industries, manufacturers and product design engineers can reap benefits from this study through continuously improving upon their products' performance.

Moreover, while LDA has been utilized in prior studies to discover product defects from textual customer feedback data, few studies have explored predictive modeling or machine learning classification using LDA results as feature vectors. As such, future research could be devoted to maximizing the usability of LDA model results by refining the performance of classification models using the distribution of topics as feature vectors. As this praxis demonstrated for dichotomous problems (i.e., "recalled" versus "not recalled"), this approach performs successfully and could be applied within other contexts.

This praxis has demonstrated various ways in which combinations of machine learning models could be utilized to detect early warning signs of medical device defects and mitigate the potential for recalls. However, the ways in which these objectives could be further explored are endless, as the field of machine learning is consistently evolving and the need for novel applications only continues to grow.

References

Abrahams, A. S., Fan, W., Wang, G. A., Zhang, Z., & Jiao, J. (2015). An integrated text analytic framework for product defect discovery. *Production and Operations Management, 24*(6), 975-990. doi: https://doi-org.proxygw.wrlc.org/10.1111/poms.12303

Abuhav, I. (2018). *ISO 13485: 2016: a complete guide to quality management in the medical device industry*. CRC Press.

Aizawa, A. (2003). *An information-theoretic perspective of tf–idf measures* Elsevier BV. doi:10.1016/s0306-4573(02)00021-3

Akarsh, S., Simran, K., Poornachandran, P., Menon, V. K., & Soman, K. P. (2019). Deep learning framework and visualization for malware classification. Paper presented at the 1059-1063. doi:10.1109/ICACCS.2019.8728471 Retrieved from https://ieeexplore.ieee.org/document/8728471

Al-Omari, O. (2019). Enhanced document classification using noun verb (Nv) terms extraction approach. International Journal of Advanced Trends in Computer Science and Engineering. 8. 85-92. 10.30534/ijatcse/2019/15812019

Antonini, et al. (2021). *A crisis-responsive framework for medical device development applied to the COVID-19 pandemic* Frontiers Media SA. doi:10.3389/fdgth.2021.617106

Argyrou, A., Giannoulakis, S., & Tsapatsoulis, N. (2018). Topic modelling on instagram hashtags: An alternative way to automatic image annotation? Paper presented at

the 61-67. doi:10.1109/SMAP.2018.8501887 Retrieved from https://ieeexplore.ieee.org/document/8501887

Asmussen, C. B., & Møller, C. (2019). Smart literature review: A practical topic modelling approach to exploratory literature review. Journal of Big Data, 6(1), 1-18. doi:10.1186/s40537-019-0255-7

Badnjević, A., Pokvić, L. G., Džemić, Z., & Bečić, F. (2020). Risks of emergency use authorizations for medical products during outbreak situations: A COVID-19 case study. *Biomedical Engineering Online, 19*(1), 75. doi:10.1186/s12938-020-00820-0

Ball, G. P., Shah, R., & Donohue, K. (2018). The decision to recall: A behavioral investigation in the medical device industry. Journal of Operations Management, 62(1), 1-15. doi:10.1016/j.jom.2018.07.003

Barbarash, R. A., Smith, L. A., Godwin, J. E., & Sahn, S. A. (1990). Mechanical Ventilation. DICP, 24(10), 959–970. https://doi.org/10.1177/106002809002401011

based on fine-tuning hyper-parameters approach. *International Journal of Computer Science and Information Security (IJCSIS), 16*(12), 155-160. Retrieved from https://arxiv.org/pdf/1902.06542.pdf

Bhat, S. K., & Culotta, A. (2017). Identifying leading indicators of product recalls from online reviews using positive unlabeled learning and domain adaptation. Retrieved from https://arxiv.org/abs/1703.00518

Billingham, S., Widrick, R., Edwards, N. J., & Klaus, S. A. (2020). *COVID-19 (SARS-CoV-2) ventilator resource management using a network optimization model and predictive system demand* Cold Spring Harbor Laboratory. doi:10.1101/2020.05.26.20113886

Blake, K. (2013). Postmarket surveillance of medical devices: Current capabilities and future opportunities.*Journal of Interventional Cardiac Electrophysiology, 36*(2), 119-127. doi:10.1007/s10840-013-9778-6

Bleaney, G., Kuzyk, M., Man, J., Mayanloo, H., & Tizhoosh, H. R. (2018). Auto-detection of safety issues in baby products. *Recent trends and future technology in applied intelligence* (pp. 505-516). Cham: Springer International Publishing. doi:10.1007/978-3-319-92058-0_49 Retrieved from http://link.springer.com/10.1007/978-3-319-92058-0_49

Blei D., Ng A., & Jordan, M. (2003). Latent dirichlet allocation. J. Mach. Learn. Res.3, null (3/1/2003), 993–1022.

Bottou, L. (2010). *Proceedings of COMPSTAT'2010*. Heidelberg: Physica-Verlag HD. doi:10.1007/978-3-7908-2604-3 Retrieved from http://link.springer.com/10.1007/978-3-7908-2604-3

Breiman, L., Last, M., & Rice, J. (2001). Random forests: Finding quasars. *Statistical challenges in astronomy*(pp. 243-254). New York, NY: Springer New York. doi:10.1007/0-387-21529-8_16 Retrieved from http://link.springer.com/10.1007/0-387-21529-8_16

Bühler, C., Hoelper, R., Hoyer, H., & Humann, W. (1995). Autonomous robot technology for advanced wheelchair and robotic aids for people with disabilities. *Robotics and Autonomous Systems, 14*(2), 213-222. doi:https://doi.org/10.1016/0921-8890(95)00030-J

Caceres, V. (2014). Clearing the shelves: Insights on how to handle medical device recalls. *Biomedical Instrumentation & Technology, 48*(6), 414-422. doi:10.2345/0899-8205-48.6.414

Callahan, A. et al. (2019). Medical device surveillance with electronic health records. *NPJ Digital Medicine, 2*(1), 94. doi:10.1038/s41746-019-0168-z

Center for Devices and Radiological Health. (2020). *Manufacturer and user facility device experience database - (MAUDE)*. U.S. Food and Drug Administration. Retrieved November 4, 2021, from https://www.fda.gov/medical-devices/mandatory-reporting-requirements-manufacturers-importers-and-device-user-facilities/manufacturer-and-user-facility-device-experience-database-maude

Center for Devices and Radiological Health. (2020). Recalls, corrections and removals (devices). Retrieved November 03, 2021, from https://www.fda.gov/medical-devices/postmarket-requirements-devices/recalls-corrections-and-removals-devices

Center for Devices and Radiological Health. (2020). *Ventilators and ventilator accessories EUAs*. U.S. Food and Drug Administration. Retrieved September 28, 2021, from https://www.fda.gov/medical-devices/coronavirus-disease-2019-

covid-19-emergency-use-authorizations-medical-devices/ventilators-and-ventilator-accessories-euas#ventilators

Center for Devices and Radiological Health. (2021). *Certain Philips Respironics ventilators, BiPAP, CPAP machines recalled*. U.S. Food and Drug Administration. Retrieved September 28, 2021, from https://www.fda.gov/medical-devices/safety-communications/certain-philips-respironics-ventilators-bipap-and-cpap-machines-recalled-due-potential-health-risks

Center for Devices and Radiological Health. (2022). *Bellavista 1000 and 1000E series ventilators recalled*. U.S. Food and Drug Administration. Retrieved from https://www.fda.gov/medical-devices/medical-device-recalls/vyaire-medical-recalls-bellavista-1000-and-1000e-series-ventilators-due-issues-software

Center for Devices and Radiological Health. (n.d.). *About manufacturer and User Facility Device Experience (Maude)*. U.S. Food and Drug Administration. Retrieved October 16, 2021 from https://www.fda.gov/medical-devices/mandatory-reporting-requirements-manufacturers-importers-and-device-user-facilities/manufacturer-and-user-facility-device-experience-database-maude

Center for Devices and Radiological Health. (n.d.). *Medical device recalls*. U.S. Food and Drug Administration. Retrieved from https://www.fda.gov/medical-devices/medical-device-safety/medical-device-recalls

Center for Devices and Radiological Health. (n.d.). Premarket Approval (PMA). Retrieved from https://www.fda.gov/medical-devices/premarket-submissions/premarket-approval-pma#ref

Cerrito, P. B. (2007). Mining the electronic medical record to examine physician decisions. Advanced computational intelligence paradigms in healthcare – 1 (pp. 113-126). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-47527-9_5 Retrieved from http://link.springer.com/10.1007/978-3-540-47527-9_5

Chang, C. et al. (2019). Effect of black tea and green tea on the radial pulse spectrum in healthy humans. *The Journal of Alternative and Complementary Medicine (New York, N.Y.), 25*(5), 559-561. doi:10.1089/acm.2018.0455

Chatburn R.L., & Mireles-Cabodevila E (2013). Chapter 3. basic principles of ventilator design. Tobin M.J.(Ed.), Principles and Practice of Mechanical Ventilation, 3e. McGraw Hill. https://accessmedicine-mhmedical-com.proxygw.wrlc.org/content.aspx?bookid=520&sectionid=41692239

Chen, S., Santoso, A., Lee, Y., & Wang, J. (2015). Latent dirichlet allocation based blog analysis for criminal intention detection system. 2015 International Carnahan Conference on Security Technology (ICCST), 73-76.

Chen, Y., Ganesan, S., & Liu, Y. (2009). Does a firm's product-recall strategy affect its financial value? an examination of strategic alternatives during product-harm crises. *Journal of Marketing, 73*(6), 214-226. doi:10.1509/jmkg.73.6.214

Clemente, F., Faiella, G., Rutoli, G., Bifulco, P., Romano, M., & Cesarelli, M. (2019). Critical failures in the use of home ventilation medical equipment. Heliyon, 5(12), e03034. https://doi.org/10.1016/j.heliyon.2019.e03034

Cohen, R., & Ruths, D. (2021). Classifying Political Orientation on Twitter: It's Not Easy!. *Proceedings of the International AAAI Conference on Web and Social Media*, *7*(1), 91-99. Retrieved from https://ojs.aaai.org/index.php/ICWSM/article/view/14434

Dalal, M., & Zaveri, M. (2011). Automatic text classification: A technical review. *International Journal of Computer Applications, 28*(2), 37-40. doi:10.5120/3358-4633

Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis, 1*(3), 131-156. doi:10.3233/IDA-1997-1302

Deerwester, S. et al. (1990). Indexing by latent semantic analysis Wiley. doi:10.1002/(sici)1097-4571(199009)41:6<391::aid-asi1>3.0.co;2-9

Diab, S. (2018). Optimizing stochastic gradient descent in text classification

Dietterich, T. (2000). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning, 40*(2), 139-157. doi:10.1023/A:1007607513941

DiMaggio, P., Nag, M., & Blei, D. (2013). Exploiting affinities between topic modeling and the sociological perspective on culture: Application to newspaper coverage of

U.S. government arts funding. Poetics (Amsterdam), 41(6), 570-606. doi:10.1016/j.poetic.2013.08.004

El-Koka, A., & Kang, D. (2016). Regularization parameter tuning optimization approach in logistic regression. Indian Journal of Science and Technology, 9(46) doi:10.17485/ijst/2016/v9i46/107857

Emakhu, J., Aguwa, C., Monplaisir, L. & Arslanturk, S. (2020). Failure Type Prediction in Software-related Medical Device Recalls. *IIE Annual Conference. Proceedings*, 628–633.

Ensign, L. G., & Cohen, K. B. (2017). A primer to the structure, content and linkage of the FDA's manufacturer and user facility device experience (MAUDE) files. *EGEMS (Washington, DC), 5*(1), 12. Retrieved from https://www.ncbi.nlm.nih.gov/pubmed/29930960

Erdoğan, H., et al. (2005). Multi-modal person recognition for vehicular applications Springer Berlin Heidelberg. doi:10.1007/11494683_37

Farre, R., et al. (2005). Quality control of equipment in home mechanical ventilation: A european survey. The European Respiratory Journal, 26(1), 86-94. doi:10.1183/09031936.05.00066904

Fong, T. H. Y., Sarkani, S., & Fossaceca, J. (2021). Auto defect detection using customer reviews for product recall insurance analysis. *Frontiers in Applied Mathematics and Statistics, 7* doi:10.3389/fams.2021.632847

Fu, Z. et al. (2017). Study of software-related causes in the FDA medical device recalls. Paper presented at the - *2017 22nd International Conference on Engineering of Complex Computer Systems (ICECCS),* 60-69. doi:10.1109/ICECCS.2017.20 Retrieved from https://ieeexplore.ieee.org/document/8292803

Geletta, S., Follett, L., & Laugerman, M. (2019). Latent dirichlet allocation in predicting clinical trial terminations. *BMC Medical Informatics and Decision Making, 19*(1), 242. doi:10.1186/s12911-019-0973-y

Gerber, M. S. (2014). Predicting crime using twitter and kernel density estimation. Decision Support Systems, 61, 115-125. doi:10.1016/j.dss.2014.02.003

Gethers, M. & Poshyvanyk, D. (2010). Using relational topic models to capture coupling among classes in object-oriented software systems. Paper presented at the - 2010 IEEE International Conference on Software Maintenance, 1-10. doi:10.1109/ICSM.2010.5609687

Giacinto, G., & Roli, F. (1997). Ensembles of Neural Networks for Soft Classification of Remote Sensing Images.

Greene, D., & Cross, J. (2015). Unveiling the political agenda of the european parliament plenary. Paper presented at the 1-10. doi:10.1145/2786451.2786464 Retrieved from http://dl.acm.org/citation.cfm?id=2786464

Grimmer, J., & Stewart, B. M. (2013). Text as data: The promise and pitfalls of automatic content analysis methods for political texts. Political Analysis, 21(3), 267-297. doi:10.1093/pan/mps028

Gupta, G., & Malhotra, S. (2015). Text document tokenization for word frequency count using rapid miner (taking resume as an example). *International Journal of Computer Applications, 975*(8887)

Gupta, G., Sharma, M., Choudhary, S., & Pandey, K. (2021). Performance analysis of machine learning classification algorithms for breast cancer diagnosis. Paper presented at the - 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 1-6. doi:10.1109/ICRITO51393.2021.9596230 Retrieved from https://ieeexplore.ieee.org/document/9596230

Hani, S., & Marcellis-Warin, N.. (2016). Open innovation and involvement of end-users in the medical device technologies' design & development process: End-users' perspectives. *Ji Shu Yu Tou Zi, 7*(3), 73-85. doi:10.4236/ti.2016.73010

Hearst, M. A. (1999). Untangling text data mining Association for Computational Linguistics. doi:10.3115/1034678.1034679

Hegde, V., & Konakanchi, K. (Jan 2011). Case study - post market product monitoring system. Paper presented at the 1-5. doi:10.1109/RAMS.2011.5754520 Retrieved from https://ieeexplore.ieee.org/document/5754520

Hobbs, J., Walker, D., Amsler, R. (1982). "Natural language access to structured text". Proceedings of the 9th conference on Computational linguistics. Vol. 1. pp. 127–32. doi:10.3115/991813.991833.

Hofmann, T. (1999). Probabilistic latent semantic indexing. doi:10.1145/312624.312649 Retrieved from http://dl.acm.org/citation.cfm?id=312649

Hora, M., Bapuji, H., & Roth, A. V. (2011). Safety hazard and time to recall: The role of recall strategy, product defect type, and supply chain player in the U.S. toy industry. *Journal of Operations Management, 29*(7), 766-777. doi:10.1016/j.jom.2011.06.006

Indra, S. T., Wikarsa, L., & Turang, R. (2016). Using logistic regression method to classify tweets into the selected topics. Paper presented at the 385-390. doi:10.1109/ICACSIS.2016.7872727 Retrieved from https://ieeexplore.ieee.org/document/7872727

Islam, T., & Goldwasser, D. (2020). Does yoga make you happy? analyzing twitter user happiness using textual and temporal information. Paper presented at the 4241-4249. doi:10.1109/BigData50022.2020.9378461 Retrieved from https://ieeexplore.ieee.org/document/9378461

Japkowicz, N. (2006). Why question machine learning evaluation methods? (an illustrative review of the shortcomings of current methods). Paper presented at the *AAAI Workshop on Evaluation Methods for Machine Learning,* 6-11.

Japkowicz, N., Ma, & He, H. (2013). Imbalanced learning : foundations, algorithms, and applications. IEEE Press. https://doi.org/10.1002/9781118646106

Jelodar, H., et al. (2019). Latent dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications, 78*(11), 15169-15211. doi:10.1007/s11042-018-6894-4

Jiang, Z. et al. (2012). Using link topic model to analyze traditional Chinese Medicine Clinical symptom-herb regularities. 2012 IEEE 14th International Conference on

e-Health Networking, Applications and Services (Healthcom), 15–18. https://doi.org/10.1109/HealthCom.2012.6380057

Kaplan, & Vakili, K. (2015). The double-edged sword of recombination in breakthrough innovation. *Strategic Management Journal*, *36*(10), 1435–1457. https://doi.org/10.1002/smj.2294

Khanal, D. (2018). Analyses of medical device failures related to computing technology. [Unpublished doctoral dissertation]. Rutgers, the State University of New Jersey.

Khoury, A. et al. (2014). From mouth-to-mouth to bag-valve-mask ventilation: evolution and characteristics of actual devices--a review of the literature. BioMed research international, 2014, 762053. https://doi.org/10.1155/2014/762053

Kini, O., Shenoy, J., & Subramaniam, V. (2013). On the determinants, financial and operating consequences, and the product market effects of product recalls. *SSRN Electronic Journal,* doi:10.2139/ssrn.2246060

Kowsari, K. et al. (2019). Text classification algorithms: A survey. *Information (Basel), 10*(4), 150. doi:10.3390/info10040150

Krishna Kumar, B., Ravi, M., Dinesh, K., & Nanda, A. (2011). Ventilator malfunction. *Journal of anesthesiology, clinical pharmacology*. 27(4), 576. https://doi.org/10.4103/0970-9185.86623

Lee, S., Kim, J., & Myaeng. S. (2015). An extension of topic models for text classification: A term weighting approach. Paper presented at the 217-224. doi:10.1109/35021BIGCOMP.2015.7072834 Retrieved from

https://ieeexplore.ieee.org/document/7072834

Li, Y., & Li, Y. (2020). Study of merchant adoption in mobile payment system based on ensemble learning. *American Journal of Industrial and Business Management, 10*(5), 861-875. doi:10.4236/ajibm.2020.105058

Liddy, E. D. (1990). *Anaphora in natural language processing and information retrieval* Elsevier BV. doi:10.1016/0306-4573(90)90008-p

Lim, K. & Park, J. (2020). Part-of-speech tagging using multiview learning. *IEEE Access, 8*, 1. doi:10.1109/ACCESS.2020.3033979

Linstead, E., Rigor, P., Bajracharya, S., Lopes, C., & Baldi, P. (2007). Mining concepts from code with probabilistic topic models. Paper presented at the 461-464. doi:10.1145/1321631.1321709 Retrieved from http://dl.acm.org/citation.cfm?id=1321709

Livingston, A. N., & Mattingly, T. J. (2021). Drug and medical device product failures and the stability of the pharmaceutical supply chain. *Journal of the American Pharmacists Association, 61*(1), e119-e122. doi:10.1016/j.japh.2020.07.005

Lohr, S. (2012). The age of big data. *New York Times*. 11.

Lui, M., Lau, J.H., & Baldwin, T. (2014). Automatic Detection and Language Identification of Multilingual Documents. *Transactions of the Association for Computational Linguistics, 2*, 27-40.

Ma, Y., & He, H. (2013). Imbalanced learning : Foundations, algorithms, and applications. John Wiley & Sons, Incorporated.

Madekurozwa, M. et al. (2021). A novel ventilator design for COVID-19 and resource-limited settings. Frontiers Media SA. doi:10.3389/fmedt.2021.707826

Markiewicz et al. (2017). Early assessment of medical devices in development for company decision making: An exploration of best practices. *Journal of Commercial Biotechnology, 23*(2), 15-30. doi:10.5912/jcb780

Mehrotra, S., Rahimian, H., Barah, M., Luo, F., & Schantz, K. (2020). A model of supply-chain decisions for resource sharing with an application to ventilator allocation to combat COVID-19. *Naval Research Logistics, 67*(5), 303-320. doi:10.1002/nav.21905

Mimno, D., & Blei, D.M. (2011). Bayesian checking for topic models. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Edinburgh, pp. 227–237.

Mishra, A., & Vishwakarma, S. (2015). Analysis of TF-IDF model and its variant for document retrieval. Paper presented at the 772-776. doi:10.1109/CICN.2015.157 Retrieved from https://ieeexplore.ieee.org/document/7546200

Money, A. G et al. (2011). The role of the user within the medical device design and development process: Medical device manufacturers' perspectives. *BMC Medical Informatics and Decision Making, 11*(1), 15. doi:10.1186/1472-6947-11-15

Mukherjee, U. K., & Sinha, K. K. (2018). Product recall decisions in medical device supply chains: A big data analytic approach to evaluating judgment bias. *Production and Operations Management, 27*(10), 1816-1833. doi:10.1111/poms.12696

Nuser, M., & Al-Horani, E. (2020). Medical documents classification using topic modeling. *Indonesian Journal of Electrical Engineering and Computer Science, 17*(3), 1524. doi:10.11591/ijeecs.v17.i3.pp1524-1530

Office of the Commissioner. (2021). Emergency use authorization. Retrieved November 03, 2021, from https://www.fda.gov/emergency-preparedness-and-response/mcm-legal-regulatory-and-policy-framework/emergency-use-authorization

Oza, N. C., & Tumer, K. (2008). Classifier ensembles: Select real-world applications. *Information Fusion, 9*(1), 4-20. doi:10.1016/j.inffus.2007.07.002

Pal, M. (2005). Random forest classifier for remote sensing classification. *International Journal of Remote Sensing, 26*(1), 217-222. doi:10.1080/01431160412331269698

Park, A. (2022). Philips, still hobbled by expanded ventilator recall, braces for Q4 earnings miss. Retrieved February 28, 2022, from https://www.fiercebiotech.com/medtech/philips-still-hobbled-by-expanded-ventilator-recall-braces-for-q4-earnings-miss

Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in python. JMLR 12(85). pp. 2825-2830.

Pesaranghader, A., Pesaranghader, A., & Rezaei, A. (2013). Applying Latent Semantic Analysis to Optimize Second-order Co-occurrence Vectors for Semantic Relatedness Measurement. *MIKE*.

Philips. (2021) Philips' second-quarter results. Retrieved from https://www.philips.com/a-

w/about/news/archive/corpcomms/news/press/2021/philips-second-quarter-results-2021.html

Prabhakar, S. K., & Won, D. (2021). Medical text classification using hybrid deep learning models with multihead attention. Computational Intelligence and Neuroscience, 2021, 1-16. doi:10.1155/2021/9425655

Pranckevičius, T., & Marcinkevičius, V. (2017). Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification. *Baltic Journal of Modern Computing, 5*(2), 221. doi:10.22364/bjmc.2017.5.2.05

R. Jayaraman, F. AlHammadi, & M. C. E. Simsekler. (2018). Managing product recalls in healthcare supply chain. Paper presented at the - *2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM),* 293-297. doi:10.1109/IEEM.2018.8607403

Raschka, S. (2014). *Naive bayes and text classification I - introduction and theory*. Unpublished. doi:10.13140/2.1.2018.3049

Redberg, R. F., & Dhruva, S. S. (2011). Medical device recalls: Get it right the first time: Comment on "Medical device recalls and the FDA approval process". *Archives of Internal Medicine (1960), 171*(11), 1011-1012. doi:10.1001/archinternmed.2011.27

Resnic, F. S. et al. (2010). Automated surveillance to detect postprocedure safety signals of approved cardiovascular devices. *JAMA : The Journal of the American Medical Association, 304*(18), 2019-2027. doi:10.1001/jama.2010.1633

Rosid, M. A., Fitrani, A. S., Astutik, I. R. I., Mulloh, N. I., & Gozali, H. A. (2020). Improving text preprocessing for student complaint document classification using sastrawi. IOP Conference Series. Materials Science and Engineering, 874(1), 12017. doi:10.1088/1757-899X/874/1/012017

Sagi, O., & Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery, 8*(4), n/a. doi:10.1002/widm.1249

Saif, H., He, Y., & Alani, H. (2018). Semantic sentiment analysis of twitter. *The semantic web – ISWC 2012* (pp. 508-524). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-35176-1_32 Retrieved from http://link.springer.com/10.1007/978-3-642-35176-1_32

Salton, G., & McGill, M. (1983). Introduction to Modern Information Retrieval.

Sapountzoglou, N., Lago, J., & Raison, B. (2020). Fault diagnosis in low voltage smart distribution grids using gradient boosting trees. *Electric Power Systems Research, 182*, 106254. doi:10.1016/j.epsr.2020.106254

Sarkissian, A. (2018). *An exploratory analysis of U.S. FDA class I medical device recalls: 2014–2018*Informa UK Limited. doi:10.1080/03091902.2019.1580778

Sbalchiero, S., & Eder, M. (2020). Topic modeling, long texts and the best number of topics. some problems and solutions. *Quality & Quantity, 54*(4), 1095-1108. doi:10.1007/s11135-020-00976-w

Scanlon, M. C., Karsh, B., & Densmore, E. M. (2006). Human factors engineering and patient safety. *The Pediatric Clinics of North America, 53*(6), 1105-1119. doi:10.1016/j.pcl.2006.09.012

Sebastiani, F. (2002). Machine learning in automated text categorization. ACM Computing Surveys, 34(1), 1-47. doi:10.1145/505282.505283

SelectUSA. (n.d.). Medical Technology Spotlight: The Medical Technology Industry in the United States. U.S. Department of Commerce. https://www.selectusa.gov/medical-technology-industry-united-states

Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*. 28(1), pp. 11-21. https://doi.org/10.1108/eb026526

Srinivasan, S. et al. (1998). Frequency, causes, and outcome of home ventilator failure. Chest, 114(5), 1363-1367. doi:10.1378/chest.114.5.1363

Stevens, K., Kegelmeyer, P., Andrzejewski, D., & Buttler, D. (2012). Exploring topic coherence over many models and many topics. In Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning (pp. 952-961).

Suffredini A. F. (2015). Caring for Critically Ill Ebola Virus Disease Patients With One Hand Tied Behind Your Back. Critical care medicine, 43(10), 2249–2250. https://doi.org/10.1097/CCM.0000000000001221

Svetnik, V., Liaw, A., Tong, C., & Wang, T. (2004). Application of breiman's random forest to modeling structure-activity relationships of pharmaceutical molecules. Multiple classifier systems (pp. 334-343). Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-540-25966-4_33 Retrieved from http://link.springer.com/10.1007/978-3-540-25966-4_33

Toman, M., Tesar, R., & Jezek, K. (2006). Influence of word normalization on text classification. *Proceedings of InSciT, 4*, 354-358.

U.S. Food and Drug Administration. (2020). *FDA at a glance regulated products and facilities*. Retrieved from https://www.fda.gov/media/143704/download.

Uysal, A. K., & Gunal, S. (2014). The impact of preprocessing on text classification. *Information Processing & Management, 50*(1), 104-112. doi:10.1016/j.ipm.2013.08.006

Vabalas, A., Gowen, E., Poliakoff, E., & Casson, A. J. (2019). Machine learning algorithm validation with a limited sample size. *PloS One, 14*(11), e0224365. doi:10.1371/journal.pone.0224365

Van Der Maaten, L., Postma, E., & Van den Herik, J. (2009). Dimensionality reduction: A comparative. *J Mach Learn Res, 10*(66-71), 13.

Venkataraman, G. et al. (2020). FasTag: Automatic text classification of unstructured medical narratives

Vockley, M. (2015). Keeping an eye on medical devices postmarket surveillance enters new age.*Biomedical Instrumentation & Technology, 49*(6), 378-392. doi:10.2345/0899-8205-49.6.378

Vulic, I., Smet, W.D., & Moens, M. (2011). Identifying Word Translations from Comparable Corpora Using Latent Topic Models. *ACL*.

West, J. B. (2005). Historical perspective the physiological challenges of the 1952 Copenhagen poliomyelitis epidemic and a renaissance in clinical respiratory physiology. *Journal of Applied Physiology*. l99: 424–432, 2005. doi:10.1152/japplphysiol.00184.2005.8750-7587/05

Xu, S. (2018). Bayesian naïve bayes classifiers to text classification. *Journal of Information Science, 44*(1), 48-59. doi:10.1177/0165551516677946

Zhao, W. et. al. (2015). A heuristic approach to determine an appropriate number of topics in topic modeling. *BMC Bioinformatics, 16 Suppl 13*(S13), S8. doi:10.1186/1471-2105-16-S13-S8

Zheng, L., He, Z., & He, S. (2020). A novel probabilistic graphic model to detect product defects from social media data. *Decision Support Systems, 137*, 113369. doi:10.1016/j.dss.2020.113369

APPENDIX A

MAUDE DATABASE DOWNLOADABLE FILES, FILE SIZE, RECORDS COUNT OF EACH, AND DESCRIPTION

(Retrieved March 1st, 2022 from https://www.fda.gov/medical-devices/mandatory-reporting-requirements-manufacturers-importers-and-device-user-facilities/manufacturer-and-user-facility-device-experience-database-maude)

File Name	Compressed Size in Bytes	Uncompressed Size in Bytes	Total Records	Description
mdrfoi.zip	9281KB	93847KB	282,730	MAUDE Base records received to date for 2021
mdrfoithru2020.zip	393087KB	3601648KB	10,800,980	Master Record through 2020
mdrfoiadd.zip	4130KB	41757KB	126,030	New MAUDE Base records for the current month.
mdrfoichange.zip	12281KB	121309KB	354,791	MAUDE Base data updates: changes to existing Base data.
patient.zip	764KB	7693KB	271,556	MAUDE Patient records received to date for 2021
patientthru2020.zip	42661KB	330421KB	10,803,961	Patient Record through 2020
patientadd.zip	368KB	3607KB	126,012	New MAUDE Patient records for the current month.
patientchange.zip	1097KB	10094KB	354,795	MAUDE Patient data updates: changes to

				existing Base data.
patientproblemcode.zip	91536KB	809323KB	11,815,201	Device Data for patient problem code
patientproblemdata.zip	11KB	25KB	998	Patient Problem Data
foidevthru1997.zip	6001KB	31217KB	136,917	Device Data through 1997
foidev1998.zip	3205KB	17539KB	63,440	Device Data for 1998
foidev1999.zip	2764KB	14798KB	52,880	Device Data for 1999
device2000.zip	1925KB	9894KB	53,114	Device Data for 2000
device2001.zip	2116KB	10960KB	59,073	Device Data for 2001
device2002.zip	2312KB	12829KB	70,383	Device Data for 2002
device2003.zip	2522KB	14089KB	77,946	Device Data for 2003
device2004.zip	2794KB	14782KB	82,885	Device Data for 2004
device2005.zip	3339KB	17516KB	99,770	Device Data for 2005
device2006.zip	3954KB	21072KB	120,484	Device Data for 2006
device2007.zip	4808KB	29940KB	172,204	Device Data for 2007
device2008.zip	5508KB	33651KB	195,471	Device Data for 2008
device2009.zip	6712KB	43264KB	243,109	Device Data for 2009
device2010.zip	8705KB	55457KB	304,402	Device Data for 2010
device2011.zip	11550KB	79923KB	446,875	Device Data for 2011
device2012.zip	13329KB	87131KB	487,726	Device Data for 2012
device2013.zip	17679KB	123153KB	682,274	Device Data for 2013
device2014.zip	20264KB	157957KB	863,778	Device Data for 2014
device2015.zip	23234KB	167425KB	862,586	Device Data for 2015

device2016.zip	24909KB	171997KB	868,046	Device Data for 2016
device2017.zip	28394KB	192122KB	938,454	Device Data for 2017
device2018.zip	32740KB	216394KB	1,050,210	Device Data for 2018
device2019.zip	38737KB	276012KB	1,333,418	Device Data for 2019
device2020.zip	42070KB	321855KB	1,567,369	Device Data for 2020
device.zip	6941KB	56106KB	283,098	Device Data received to date for 2021
deviceadd.zip	3209KB	25155KB	126,203	New MAUDE Device data for the current month.
devicechange.zip	8936KB	69129KB	355,274	Device data updates: changes to existing Device data and additional Device data for existing Base records.
deviceproblemcodes.zip	16KB	42KB	1,704	Device Problem Data
foidevproblem.zip	34748KB	179760KB	13,082,601	Device Data for foidevproblem
foitextthru1995.zip	3561KB	16551KB	27,401	Narrative data through 1995
foitext1996.zip	2782KB	9318KB	32,059	Narrative Data for 1996
foitext1997.zip	7557KB	26382KB	91,009	Narrative Data for 1997
foitext1998.zip	5924KB	20773KB	68,316	Narrative Data for 1998
foitext1999.zip	4501KB	15788KB	51,119	Narrative Data for 1999
foitext2000.zip	4760KB	16491KB	52,625	Narrative Data for 2000
foitext2001.zip	5090KB	17763KB	57,986	Narrative Data for 2001
foitext2002.zip	6089KB	22304KB	64,859	Narrative Data for 2002

foitext2003.zip	6221KB	23332KB	66,241	Narrative Data for 2003
foitext2004.zip	6090KB	21742KB	56,117	Narrative Data for 2004
foitext2005.zip	9649KB	34692KB	95,044	Narrative Data for 2005
foitext2006.zip	20092KB	69183KB	177,414	Narrative Data for 2006
foitext2007.zip	25177KB	88484KB	232,627	Narrative Data for 2007
foitext2008.zip	28286KB	101218KB	264,972	Narrative Data for 2008
foitext2009.zip	40869KB	147687KB	388,042	Narrative Data for 2009
foitext2010.zip	62269KB	252984KB	635,654	Narrative Data for 2010
foitext2011.zip	94583KB	425073KB	1,040,278	Narrative Data for 2011
foitext2012.zip	108957KB	475208KB	1,167,621	Narrative Data for 2012
foitext2013.zip	139507KB	622562KB	1,609,332	Narrative Data for 2013
foitext2014.zip	164295KB	788040KB	1,948,365	Narrative Data for 2014
foitext2015.zip	184834KB	861735KB	2,073,784	Narrative Data for 2015
foitext2016.zip	193641KB	918988KB	2,174,763	Narrative Data for 2016
foitext2017.zip	195288KB	974667KB	2,266,536	Narrative Data for 2017
foitext2018.zip	187882KB	961862KB	2,459,005	Narrative Data for 2018
foitext2019.zip	197295KB	1080573KB	2,810,657	Narrative Data for 2019
foitext2020.zip	188030KB	1099755KB	2,965,324	Narrative Data for 2020
foitext.zip	26424KB	159504KB	466,489	Narrative Data received to date for 2021
foitextadd.zip	11527KB	69824KB	203,841	New MAUDE Narrative data for the current month.
foitextchange.zip	33960KB	205137KB	523,899	Narrative data updates: changes

				to existing narrative data and additional narrative data for existing base records.

Appendix B

FILTER, CONCATENATE AND REMOVE DUPLICATES OF TEXT FILES (ALSO USED FOR DEVICE FILES)

```
#connect to the Google Drive
from google.colab import drive drive.mount('/content/drive/', force_remount=True)
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
# supporting libraries
import pandas as pd
#Data preview
#input folders
input_folder='/content/drive/MyDrive/filtered-foitextfiles/'
output_folder='/content/drive/MyDrive/filtered-foitextfiles/'
#input folders
input_folder='/content/drive/MyDrive/filtered-foitextfiles/'
output_folder='/content/drive/MyDrive/filtered-foitextfiles/'

file_name1 = 'filtered_foitext2012.txt'
file_name2 = 'filtered_foitext2013.txt'
file_name3 = 'filtered_foitext2014.txt'
file_name4 = 'filtered_foitext2015.txt'
file_name5 = 'filtered_foitext2016.txt'
file_name6 = 'filtered_foitext2017.txt'
file_name7 = 'filtered_foitext2018.txt'
file_name8 = 'filtered_foitext2019.txt'
# load data
df_data = pd.read_csv(input_folder + file_name, #file location
                      encoding = "ISO-8859-1", #deal with texts in different formats
                      sep='|' #column delimiter
                      )
print("Data shape:", df_data.shape)
# display first row of the data frame df_data.head(1).T
#Filtering the data
#Filter the Text Type Code column to only show Records with “D” and Patient Sequence
Number column to show only records with “1”. Finally, keep only one row for each
MDR_Report_Key by deleting all other duplicates.
#filter the Text Type Code column to only show Records with “D”
print("Data length before the filter", len(df_data))
df_data = df_data[df_data['TEXT_TYPE_CODE'] == "D"]
print("Data length after the filter", len(df_data))
#filter the Patient Sequence Number column to show only records with “1”
print("Data length before the filter", len(df_data))
```

```
df_data = df_data[df_data['PATIENT_SEQUENCE_NUMBER'] == 1]
print("Data length after the filter", len(df_data))
#delete duplicates in MDR_Report_Key keeping just one row
print("Data length before the filter", len(df_data))
df_data = df_data.drop_duplicates(subset=["MDR_REPORT_KEY"],
                          keep='first')
print("Data length after the filter", len(df_data))
#save filtered data
df_data.to_csv(output_folder + "filtered_" + file_name,
           sep="|",
           index=False)
#Concatenating several dataframes
#Merge files together to process as a single file.
df1 = pd.read_csv(input_folder + file_name1, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df2 = pd.read_csv(input_folder + file_name2, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df3 = pd.read_csv(input_folder + file_name3, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df4 = pd.read_csv(input_folder + file_name4, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df5 = pd.read_csv(input_folder + file_name5, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df6 = pd.read_csv(input_folder + file_name6, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df7 = pd.read_csv(input_folder + file_name7, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
df8 = pd.read_csv(input_folder + file_name8, #file location
                  encoding = "ISO-8859-1", #handles texts in different formats
                  sep='|' #column delimiter
                 )
```

```
#Put all dataframes into a list
list_dfs = [df1, df2, df3, df4, df5, df6, df7, df8]
#concatenate into one dataframe
df_concatenated = pd.concat(objs = list_dfs,
                                    axis=0
                                    )
#checks
print("Length of the concatenated dataframe:", len(df_concatenated))
print("Sum of lengths of dataframes that are concatenated:",
len(df1)+len(df2)+len(df3)+len(df4)+len(df5)+len(df6)+len(df7)+len(df8))
df_concatenated.head(1).T
#delete duplicates in MDR_Report_Key keeping just one row
print("Data length before the filter", len(df_concatenated))
df_concatenated = df_concatenated.drop_duplicates(subset=["MDR_REPORT_KEY"],
                          keep='first')
print("Data length after the filter", len(df_concatenated))
#save concatenated data
df_concatenated.to_csv(output_folder + "concatenated"+file_name1,
            sep="|",
            index=False)
```

APPENDIX C

PREPROCESSING DATASET

1) POS Tagging

```
#check POS of words in the corpus

df_data['list_of_POSs'] = df_data['doc'].apply(lambda x: [word.pos_ for word in x] )


#collecting all POSs in each text in a list

tmp_list = df_data['list_of_POSs'].tolist()

list_of_POSs = [POS for sublist in tmp_list for POS in sublist]


#count total frequencies of each POS in the corpus

POS_freq_counter = collections.Counter(list_of_POSs)

s_POS_freq = pd.Series(POS_freq_counter)


#sort document frequencies of each POS in the corpus

s_tmp = s_POS_freq.sort_values(ascending=False)


print('Total number of POSs: ', len(set(s_POS_freq)))


df_tmp = pd.DataFrame(s_tmp, columns=['Frequency'])


#plot df_tmp

df_tmp.plot.bar(alpha=0.5, color='c', legend=False, title='POS frequency in all texts
combined')


#Selecting POS to be used as features

selected_POSs = ['VERB', 'NOUN']
```

```
print('selecting POSs:')
selected_POSs
```

2) Lemmatization

```
# lemmatize words with selected POSs
# select lemmas that are not stop-words and have at least 3 letters
df_data['list_of_lemmas'] = df_data['doc'].apply(lambda x: [word.lemma_.lower()
for word in x if (word.is_stop==False) & \
(len(word.text)>2) & \
 (word.is_alpha) & \
 (word.pos_ in selected_POSs)])
```

3) Stop-word removal

```
#Stop-word Removal
def clean_NP(noun_phrase):
  res_np = ""

  #consider only phrases with several words to avoid duplicates
  #(single word phrases are selected as NOUNs)
  if len(noun_phrase) > 1:
    k = 0
    for word in noun_phrase:
      if (word.is_stop==False) & (len(word.lemma_)>0):
        res_np = res_np + "_" + word.lemma_.lower()
        k = k + 1
  if (len(res_np) > 0) and (k > 1):
    return res_np[1:]
  else:
```

```
    return ""
df_data['noun_phrases'] = df_data['doc'].apply(lambda x: [clean_NP(noun_phrase) for noun_phrase in list(x.noun_chunks)])


print("Examples of extracted noun phrases:")

df_data['noun_phrases'].head()

df_data['list_of_lemmas_and_NPs'] = df_data['list_of_lemmas'] + df_data['noun_phrases']


#remove zero length strings

df_data['list_of_lemmas_and_NPs'] = df_data['list_of_lemmas_and_NPs'].apply(lambda x: [word for word in x if len(word)>0])


#example of a text

ind = 0

print_long_string(df_data['EVENT_TEXT'].iloc[ind])

print('\n#Selected lemmas and noun phrases:\n')

print_long_string(str(df_data['list_of_lemmas_and_NPs'].iloc[ind]))
```

4) Checking and updating dictionary dimension (number of unique lemmas)

```
#get all unique selected lemmas from each text

tmp_list = df_data['list_of_lemmas_and_NPs'].apply(set).apply(list).tolist()

list_of_lemmas = [lemma for sublist in tmp_list for lemma in sublist]


#count lemmas' document frequencies in the corpus

lemma_freq_counter = collections.Counter(list_of_lemmas)

s_lemma_NPs_freq = pd.Series(lemma_freq_counter)


print('Total number of unique lemmas and NPs: ', len(s_lemma_NPs_freq))

print ("\nDistribution of lemmas' and NPs'' document counts: ")

print(s_lemma_NPs_freq.describe(percentiles=[0.55, 0.65, 0.75, 0.85, 0.95, 0.97, 0.99]))
```

```
#look through to 20 most/least frequent lemmas
s_tmp = s_lemma_NPs_freq.sort_values(ascending=False)
df_tmp = pd.DataFrame({'Most freq words': list(s_tmp.index[:20]),
                       'M_freq': list(s_tmp.iloc[:20]),
                       'Least freq words': list(s_tmp.index[-20:]),
                       'L_freq': list(s_tmp.iloc[-20:])})
df_tmp

#select upper and lower boundary for lemmas' count
up_bound = s_lemma_NPs_freq.quantile(0.99)
low_bound = 2

print('Lemma count upper bound:', up_bound)
print('Lemma count lower bound:', low_bound)

#select lemmas and NPs
selected = set(s_lemma_NPs_freq[(s_lemma_NPs_freq >=
low_bound)&(s_lemma_NPs_freq <= up_bound)].index)

#select lemmas in each document if they belong to chosen list of lemmas
df_data['selected_list_of_lemmas_NPs'] =
df_data['list_of_lemmas_and_NPs'].apply(lambda x:
                                        [l for l in x if l in selected])
```

5) Saving preprocessed data file

```
#list all columns
df_data.columns

#save file
```

```
with open(output_folder + 'clean_data.pickle', 'wb') as f:

    #Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(df_data[columns], f, pickle.HIGHEST_PROTOCOL)
```

6) Saving preprocessed data (subset) file

```
#list all columns
df_data.columns

#save file
with open(output_folder + 'clean_data_1.pickle', 'wb') as f:

    #Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(df_data[columns], f, pickle.HIGHEST_PROTOCOL)
```

APPENDIX D

PYTHON CODE—LDA TOPIC MODEL (ran using Google Colab)

```
#Load data and python libraries

%matplotlib inline


#Connect to the Google Drive
from google.colab import drive
drive.mount('/content/drive/', force_remount=True)


#Topic modeling libraries
import gensim
from gensim import models, corpora
from gensim.models.coherencemodel import CoherenceModel
import pyLDAvis.gensim


#Data visualization libraries
import matplotlib.pyplot as plt
import seaborn as sns


#Progress meter for loops
from tqdm.notebook import tqdm_notebook,tnrange,tqdm


#Supporting libraries
import pickle
import numpy as np
import pandas as pd
import collections
```

```
import time

pd.set_option('max_colwidth',100)

import warnings
warnings.simplefilter(action='ignore', category=DeprecationWarning)

#File location
input_folder='/content/drive/MyDrive/_Projects_/transfer_files/'
output_folder='/content/drive/MyDrive/_Projects_/transfer_files/'

file_name = 'clean_data.pickle'

#Load data
with open(input_folder + file_name, 'rb') as f:
    df_data = pickle.load(f)

#Create Bag of Words and document-term matrix for the LDA model

#Create a vocabulary for the LDA model
dictionary = corpora.Dictionary(df_data['selected_list_of_lemmas_NPs'])
#Save dictionary for classification
with open(output_folder + "LDA_dictionary.pickle", 'wb') as f:
    #Pickle the LDA dictionary using the highest protocol
    pickle.dump(dictionary, f, pickle.HIGHEST_PROTOCOL)

#Count the number of occurrences of each distinct token in each document
df_data['doc2bow'] = df_data['selected_list_of_lemmas_NPs'].apply(lambda x:
dictionary.doc2bow(x))
```

```
df_data[['selected_list_of_lemmas_NPs','doc2bow']].head()

#Create document-term matrix for LDA
doc_term_matrix = list(df_data['doc2bow'].values)

#Estimate number of topics through topic coherence

#Evaluating performance of LDA parameter "number of topics" through grid search
warnings.filterwarnings('ignore')
start_time = time.time()
LDA = models.LdaMulticore
num_topics_list = np.arange(1,25)
coherenceList_cv = []
print("Processing...")
#For each number of topics calculate Coherence Score
for num_topics in tqdm(num_topics_list):
    #LDA model
    lda= LDA(corpus=doc_term_matrix, num_topics=num_topics, id2word = dictionary,
            passes=20,chunksize=4000,random_state=3,workers=2)

    #C_v coherence score
    cm_cv = CoherenceModel(model=lda, corpus=doc_term_matrix,
                        texts=df_data['selected_list_of_lemmas_NPs'], dictionary=dictionary,
coherence='c_v')

    #Save model's coherence values for the plot
    coherenceList_cv.append(cm_cv.get_coherence())

print("Processing time in minutes:", round((time.time() - start_time)/60,2))
#Plot estimated Coherence Score by each topic
```

```
df_plot = pd.DataFrame({'Number of topics':num_topics_list,
                        'Coherence Score':coherenceList_cv})
f, ax = plt.subplots(figsize=(10, 5))
plt.style.use('seaborn-whitegrid')

sns.set_context("notebook", font_scale=1.1, rc={"lines.linewidth": 1.5})
sns.pointplot(x ='Number of topics', y = 'Coherence Score', data=df_plot)

#plt.axhline(y=0.547)

ax.grid(True, ls='--', alpha=0.5)
plt.title('Topic Coherence\n', y=1.05, fontsize=18)

#Get full table of coherence values per number of topics
coherenceList_cv

#Define LDA model
#Define the model with chosen number of topics
num_topics = 14 #selected number of topics here

LDA = models.LdaMulticore
result_lda_model= LDA(corpus=doc_term_matrix,
            num_topics=num_topics,
            id2word = dictionary,
            passes=20,
            chunksize=4000,
            random_state=3)

#Save model for future use
```

```
result_lda_model.save(output_folder + "LDA_model")


#Evaluate topics using interactive visualizations using pyLDAvis

pyLDAvis.enable_notebook()

pyLDAvis.gensim.prepare(result_lda_model, doc_term_matrix,
dictionary,sort_topics=False)


#Get words with support weights by topic

words_by_topic = [(i+1, result_lda_model.show_topic(topicid=i,topn=N)) for i in
topic_number]

words_by_topic = [(twl[0],wp[0], wp[1]) for twl in words_by_topic for wp in twl[1]]


#Add new columns to dataset

df_word_support_weight = pd.DataFrame(words_by_topic, columns=["topic",
"leading_word", "support_weight"])

df_word_support_weight


#Export full dataset to output folder

df_word_support_weight.to_csv(output_folder + "topics_support_weights.csv")
```

APPENDIX E

PYTHON CODE—RECALL STATUS CLASSIFICATION USING EXTRACTED EARLY WARNING SIGNS OF DEFECTS FROM LDA MODEL (ran using Google Colab)

```
!python -m spacy download en_core_web_md

#Connect to the Google Drive
from google.colab import drive
drive.mount('/content/drive/', force_remount=True)

#Import all libraries
#General
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import pickle
#Text processing libraries
import re
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
import en_core_web_md
#Initialize spacy model, keeping only tagger component (for efficiency)
nlp = en_core_web_md.load(disable=['ner'])
#Topic modeling libraries
import gensim
from gensim import models, corpora
```

```
from gensim.models.coherencemodel import CoherenceModel

#Data visualization libraries

import matplotlib

import matplotlib.pyplot as plt

import seaborn as sns

#Features

from sklearn.preprocessing import StandardScaler

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.preprocessing import LabelEncoder, label_binarize
#Classifier models
from sklearn.linear_model import SGDClassifier

from sklearn.naive_bayes import GaussianNB

from sklearn.tree import DecisionTreeClassifier

from sklearn.linear_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier

from sklearn.metrics import classification_report

#Model performance metrics/evaluation

from sklearn.model_selection import train_test_split, GridSearchCV

from sklearn.metrics import confusion_matrix, accuracy_score, f1_score

from sklearn.preprocessing import label_binarize

from scipy import interp

from itertools import cycle

#Measure run-time

import time

start_time = time.time()


#Define topics in the text based on pretrained model

#supporting functions

def get_LDA_model(path):
```

```
    lda_model = models.LdaMulticore.load(path)
    return lda_model
def get_LDA_dictionary(path):
    with open(path, 'rb') as f:
        dictionary = pickle.load(f)
    return dictionary
def get_list_of_lemmas_and_NPs(text):
  #Repeat text preprocessing step
  selected_POSs = ['VERB', 'NOUN']
  doc = nlp(text)
  #Lemmatize words with selected POSs, select lemmas that are not stop-words
  list_of_lemmas = [word.lemma_.lower() for word in doc if (word.is_stop==False) & \
                                        (len(word.text)>2) & \
                                        (word.is_alpha) & \
                                        (word.pos_ in selected_POSs)]
  def clean_NP(noun_phrase):
    res_np = ""
    #Consider only phrases with several words to avoid duplicates
    #(single word phrases are selected as NOUNs)
    if len(noun_phrase) > 1:
      k = 0
      for word in noun_phrase:
        if (word.is_stop==False) & (len(word.lemma_)>0):
          res_np = res_np + "_" + word.lemma_.lower()
          k = k + 1
    if (len(res_np) > 0) and (k > 1):
      return res_np[1:]
    else:
      return ""
  noun_phrases = [clean_NP(noun_phrase) for noun_phrase in list(doc.noun_chunks)]
```

```
  #combine two lists
  list_of_lemmas_and_NPs = list_of_lemmas + noun_phrases
  #remove zero length strings
 list_of_lemmas_and_NPs = [word for word in list_of_lemmas_and_NPs if len(word)>0]

  return list_of_lemmas_and_NPs

def get_all_topics(text, LDA_dictionary, LDA_model):
  """
  Returns list of tuples (topic, topic score)
  sorted in descending order
  """
  #Get doc2bow for the new text
  new_list_of_lemmas_and_NPs = get_list_of_lemmas_and_NPs(text)
  doc2bow = LDA_dictionary.doc2bow(new_list_of_lemmas_and_NPs)

  result = sorted(LDA_model[doc2bow], key=lambda tup: -1*tup[1])
  result = [(t[0]+1, t[1]) for t in result]

  return result

#Specify file location
input_folder='/content/drive/MyDrive/_Projects_/transfer_files/'
output_folder='/content/drive/MyDrive/_Projects_/output/'
#Download pre-trained LDA model
LDA_model = get_LDA_model(input_folder + "LDA_model")
LDA_dictionary = get_LDA_dictionary(input_folder + "LDA_dictionary.pickle")

#Find topics for each text in the data
```

```
start_time = time.time()

file_folder='/content/drive/MyDrive/_Projects_/'

file_name = 'file_name.txt'

#Load data

df_data = pd.read_csv(file_folder + file_name,

                      encoding = "ISO-8859-1",

                      sep='\t'

                     )

df_data = df_data[[‘MDR_REPORT_KEY’, 'RECALL_STATUS', 'EVENT_TEXT']]

#find topics for texts in a column

column = "EVENT_TEXT"

df_data['topics'] = df_data[column].fillna("").apply(lambda text: get_all_topics(text,
LDA_dictionary, LDA_model))


print("Processing time in minutes:", round((time.time() - start_time) / 60, 2))
print("Data shape:", df_data.shape)


#Prepare data for classification

#Get text length

df_data['EVENT_TEXT - len'] = df_data['EVENT_TEXT'].fillna("").apply(len)


#Get all topics in a dictionary with 0 values

df_data['topics_num'] = df_data['topics'].apply(lambda x: [t[0] for t in x])

list_of_topics = list(df_data['topics_num'].values)

list_of_topics = [t for l in list_of_topics for t in l]

list_of_topics = list(set(list_of_topics))

list_of_topics.sort()

topics_dict = dict(zip(list_of_topics, [0]*len(list_of_topics)))

topics_dict
```

```
#For each row in the df_data get dictionary of topics with values equal to topic scores
def fill_topic_dict(topics):
  global topics_dict

  dict_tmp = topics_dict.copy()

  for (t,s) in topics:
    dict_tmp[t] = s
  return dict_tmp

df_data['topics_dict'] = df_data['topics'].apply(lambda x: fill_topic_dict(x))

#Add breakdown of topic distributions per row
list_tmp = list(df_data['topics_dict'])
df_tmp = pd.DataFrame(list_tmp)
df_tmp = df_tmp.set_axis(["Topic_"+str(t) for t in df_tmp.columns], axis=1,
inplace=False)
topic_columns = list(df_tmp.columns)
df_data[topic_columns] = df_tmp
df_data.tail().T
df_data

#Create Labels for Classification Model

#Creating classification label
label_column='label'

df_data[label_column] = 'RECALL_STATUS : ' +
df_data['RECALL_STATUS'].apply(str)
df_data[label_column].value_counts()
```

```
#Convert labels into numbers for classification
LE = LabelEncoder()
df_data[label_column + '_num'] = LE.fit_transform(df_data[label_column])

s_labels = df_data.groupby(label_column)[label_column + '_num'].min()
dict_product_id = s_labels.to_dict()
pd.DataFrame(s_labels)

#Classification Model Run
#Split the dataset with stratification to maintain label proportions
df_X_train, df_X_test, s_y_train, s_y_test = train_test_split(df_data[['EVENT_TEXT -
len'] + topic_columns],
                                                df_data[label_column + '_num'],
                                                test_size = 0.2, stratify = df_data['label_num'],
random_state = 101)

#Check label proportions in training set split
s_y_train.value_counts()

#Check label proportions in test set split
s_y_test.value_counts()

#Define the second smallest number of records with a label in the training dataset
l_counts = s_y_train.value_counts().sort_values()
sample_size = l_counts.iloc[1]
print("Second smallest number of records with a label: ", sample_size)

#Select label_size random records (without replacement) of each selected label if number
of records > label_size
```

```
df_tmp_train = df_X_train.copy()
df_tmp_train[label_column + '_num'] = s_y_train
df_train_list = []
labels = list(l_counts.index)
for label in labels:
    replace_value = False #No replacement
    df_label = df_tmp_train[df_tmp_train[label_column + "_num"] == label]
    if len(df_label) > sample_size:
      df_tmp = df_label.sample(n=sample_size, random_state=42, replace=replace_value)
      df_train_list.append(df_tmp)
    else:
      df_train_list.append(df_label)
df_train = pd.concat(df_train_list)
print('TRAIN Dataset length:', len(df_train))
print('\nLabel counts:')
print(df_train[label_column + "_num"].value_counts())

#Standardize numeric features
x_columns = ['EVENT_TEXT - len'] + topic_columns
scaler = StandardScaler().fit(df_train[x_columns].values)
X = scaler.transform(df_train[x_columns].values)
X_test = scaler.transform(df_X_test[x_columns].values)

#Data for model training
y = df_train[label_column + "_num"].values
#Data for model testing
y_test = s_y_test.values #target
print('Data shapes for model selection:')
print ('X:', X.shape)
```

```
print('y:', y.shape)
print('\nData shapes for model evaluation:')
print ('X_test:', X_test.shape)
print('y_test:', y_test.shape)

#Classification Model Runs
#Train test split with stratified sampling for evaluation
X_train, X_eval, y_train, y_eval = train_test_split(X, y,
                                    test_size = .2,
                                    shuffle = True,
                                    stratify = y,
                                    random_state = 3)

#Creating a dictionary of models for classification
model_dict = {'Stochastic Gradient Descent' : SGDClassifier(),
          'Random Forest': RandomForestClassifier(),
          'AdaBoost': AdaBoostClassifier(),
          'Gaussian Naive Bayes': GaussianNB(),
          'LogisticRegression':LogisticRegression()
          }
#Creating a dictionary of model parameters
params_SGD = dict(loss=['log'],
              penalty=['l2','l1'],
              alpha=[1e-6, 1e-3, 1e-1, 1e0],
              max_iter=[5, 1000, 10000],
              tol=[None, 1e-3],
              random_state=[3])
params_RF = dict(bootstrap = [True, False],
              max_depth = [10, 50, 100, None],
```

```
            max_features = ['auto', 'sqrt'],
            min_samples_leaf = [1, 2, 4],
            n_estimators = [500, 1000],
            random_state=[3])
params_AB = dict(base_estimator=[DecisionTreeClassifier(max_depth=1),
DecisionTreeClassifier(max_depth=2)],
            learning_rate=[1, 1.5, 2],
            n_estimators=[100, 200],
            algorithm=["SAMME", "SAMME.R"],
            random_state=[3])
parameters_dict = {'Stochastic Gradient Descent' : params_SGD,
             'Random Forest': params_RF,
             'AdaBoost': params_AB,
             'Gaussian Naive Bayes': {},
             'LogisticRegression':{}}
#Runs
#Define Scoring metric
scoring_metric = 'f1_macro'
model_name, model_parameters, model_best_score = [], [], []
model_names = model_dict.keys()

for MN in model_names:
   print("="*45)
   print(MN)
   clf = model_dict[MN]
   params = parameters_dict[MN]

   gridsearch = GridSearchCV(clf,
                        params,
                        scoring=scoring_metric,
```

```
                cv=5, #Specify number of folds in cross validation
                verbose=1,
                n_jobs=10)

    best_model = gridsearch.fit(X_train, y_train)
    model_name.append(MN)
    model_parameters.append(gridsearch.best_params_)
    model_best_score.append(gridsearch.best_score_ )

#Compare different models, get macro-f1 score and optimal parameters chosen for each
model_comparison_df = pd.DataFrame([model_name, model_best_score,
model_parameters]).T
model_comparison_df.columns = ['model_name', scoring_metric,'parameters']

model_comparison_df = model_comparison_df.sort_values(by=scoring_metric,
ascending=False).copy().reset_index()
clf_name = model_comparison_df['model_name'].iloc[0]
clf_params = model_comparison_df['parameters'].iloc[0]

print('Best model:', clf_name, clf_params)
model_comparison_df

#Model Evaluation
#Testing other classifiers' performances is done by selecting it from the table above
row_number = 0 #Default is 0, as it is the best performing model, but can be changed to
view other classifiers' results (0,1,2,3,4)
clf_name = model_comparison_df['model_name'].iloc[row_number]
clf_params = model_comparison_df['parameters'].iloc[row_number]

print('Selected model:', clf_name)
```

```
#Model Performance on Training Data
clf = model_dict[clf_name]
clf.set_params(**clf_params)
clf.fit(X_train, y_train)
y_eval_pred = clf.predict(X_eval)

print(classification_report(y_eval, y_eval_pred, digits=3)) #Performance results are
rounded to 3 significant digits

#Model Performance on Test Data
clf = model_dict[clf_name]
clf.set_params(**clf_params)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print(classification_report(y_test, y_pred, digits=3)) #Performance results are rounded to
3 significant digits

#Confusion Matrix
cm = confusion_matrix(y_test, y_pred)

cm_df = pd.DataFrame(cm,
                     index = list(s_labels),
                     columns = list(s_labels))
#Plot the heatmap
plt.figure(figsize=(10, 10))
sns.heatmap(cm_df,
            center=0,
            cmap=sns.diverging_palette(10, 240, s=80, l=55, as_cmap=True),
```

```
        annot=True,
        fmt='g')
plt.title(clf_name + '\n unseen F1-score: ' + str(f1_score(y_test, y_pred,
average="macro")), fontsize = 18)
plt.ylabel('True label', fontsize = 16)
plt.xlabel('Predicted label', fontsize = 16)
plt.show()
```

APPENDIX F

PYTHON CODE— CLASSIFICATION MODEL FOR APPLICABLE RECALL CLASS PREDICTION FROM ADVERSE EVENT NARRATIVES (ran using Google Colab)

```
!python -m spacy download en_core_web_md

#Import all libraries
#General
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
import numpy as np
import pickle
#Data visualization libraries
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
#Features
from sklearn.preprocessing import StandardScaler
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder, label_binarize
#Classifiers
from sklearn.linear_model import SGDClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.metrics import classification_report
```

```
#Metrics/evaluation
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import confusion_matrix, accuracy_score, f1_score
from sklearn.preprocessing import label_binarize
from scipy import interp
from itertools import cycle
#Measure run-time
import time
start_time = time.time()
#Text processing libraries
import re
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
import en_core_web_md
#Initialize spacy model, keeping only tagger component
nlp = en_core_web_md.load(disable=['ner'])

#Import Preprocessed Dataset
#File location
input_folder='/content/drive/MyDrive/_Projects_/transfer_files/'
output_folder='/content/drive/MyDrive/_Projects_/output/'
file_name = 'clean_data_1.pickle'

#Load data
with open(input_folder + file_name, 'rb') as f:
    df_data = pickle.load(f)
print("Data shape:", df_data.shape)
#Display first row of the data frame
df_data.tail(1).T
```

```
#Preparing data for classification
#Delete Recall_Class = N/A

df_data = df_data[df_data['RECALL_STATUS'] != 'N']

df_data.shape


#Creating classification label

label_column='label'

df_data[label_column] = 'RECALLED : ' + df_data['RECALL_STATUS'].apply(str) + "
|| RECALL CLASS:" + df_data['RECALL_STATUS_CLASSIFICATION'].apply(str)

df_data[label_column].value_counts()


#Turning labels into numbers for classification

LE = LabelEncoder()

df_data[label_column + '_num'] = LE.fit_transform(df_data[label_column])

s_labels = df_data.groupby(label_column)[label_column + '_num'].min()

dict_product_id = s_labels.to_dict()

pd.DataFrame(s_labels)


#Splitting the data with stratification to keep proportions of labels

df_X_train, df_X_test, s_y_train, s_y_test = train_test_split(df_data[['EVENT_TEXT -
len', 'list_of_lemmas_and_NPs']],

                                        df_data[label_column + '_num'],

                                        test_size = 0.3, stratify = df_data['label_num'],
random_state = 101)


#Check label proportions in training set

s_y_train.value_counts()

#Check label proportions in test set

s_y_test.value_counts()
```

```
#Define the second smallest number of records with a label in the training dataset
l_counts = s_y_train.value_counts().sort_values()
sample_size = l_counts.iloc[1]
print("Second smallest number of records with a label: ", sample_size)

#Selecting label_size random records (without replacement) of each selected label if number of records > label_size
df_tmp_train = df_X_train.copy()
df_tmp_train[label_column + '_num'] = s_y_train
df_train_list = []
labels = list(l_counts.index)
for label in labels:
    replace_value = False #No replacement
    df_label = df_tmp_train[df_tmp_train[label_column + "_num"] == label]

    if len(df_label) > sample_size:
      df_tmp = df_label.sample(n=sample_size, random_state=42, replace=replace_value)
      df_train_list.append(df_tmp)
    else:
      df_train_list.append(df_label)

df_train = pd.concat(df_train_list)

print('TRAIN Dataset length:', len(df_train))
print('\nLabel counts:')
print(df_train[label_column + "_num"].value_counts())

#Create Tf-Idf vectorizer
```

```
def text_tokenizer(text):
    return text
tfidf_vectorizer = TfidfVectorizer(stop_words=None, #Stop-words are deleted during cleaning
                                   ngram_range=(1,1), #text is preprocessed so that during tokenization we need only uni-grams
                                   max_df=0.99, #Deletes terms that have a document frequency > max_df as a proportion of all texts
                                   min_df=2,  #Terms that have a document frequency strictly lower than min_df are deleted
                                   lowercase=False,
                                   tokenizer=text_tokenizer,
                                   preprocessor=text_tokenizer
                                   )
#Standardize non Tf-Idf features
x_columns = ['EVENT_TEXT - len']

scaler = StandardScaler().fit(df_train[x_columns].values)
X_add_features = scaler.transform(df_train[x_columns].values)
X_test_add_features = scaler.transform(df_X_test[x_columns].values)

#Data for model training
tf_idf = tfidf_vectorizer.fit_transform(df_train['list_of_lemmas_and_NPs']).toarray() #Tf-Idf features
X = np.hstack((X_add_features,tf_idf)) #Additional features
y = df_train[label_column + "_num"].values #Target

#Data for model testing
tf_idf_test = tfidf_vectorizer.transform(df_X_test['list_of_lemmas_and_NPs']).toarray() #Tf-Idf features
X_test = np.hstack((X_test_add_features,tf_idf_test)) #Additional features
```

```
y_test = s_y_test.values #Target

print('Data shapes for model selection:')
print ('X:', X.shape)
print('y:', y.shape)
print('\nData shapes for model evaluation:')
print ('X_test:', X_test.shape)
print('y_test:', y_test.shape)

#Check vocabulary
v = list(tfidf_vectorizer.vocabulary_)
print("Vocabulary size: ", len(v))
print("EXAMPLES:\n")
v[::300]

#Classification Model Runs
#Train test split with stratified sampling for evaluation
X_train, X_eval, y_train, y_eval = train_test_split(X, y,
                                    test_size = .3,
                                    shuffle = True,
                                    stratify = y,
                                    random_state = 2)

#Create a dictionary of models for text classification
model_dict = {'Stochastic Gradient Descent' : SGDClassifier(),
        'Random Forest': RandomForestClassifier(),
        'AdaBoost': AdaBoostClassifier(),
        'Gaussian Naive Bayes': GaussianNB(),
        'LogisticRegression':  LogisticRegression(),
```

```
        }

#Create a dictionary of parameters for the models
params_SGD = dict(loss=['log','hinge'],
                  penalty=['l2','l1'],
                  alpha=[1e-6, 1e-3, 1e-1, 1e0],
                  max_iter=[5, 1000, 10000],
                  tol=[None, 1e-3],
                  random_state=[2])
params_RF = dict(bootstrap = [True, False],
                  max_depth = [10, 50, 100, None],
                  max_features = ['auto', 'sqrt'],
                  min_samples_leaf = [1, 2, 4],
                  n_estimators = [500, 1000],
                  random_state=[2])
params_AB = dict(base_estimator=[DecisionTreeClassifier(max_depth=1),
DecisionTreeClassifier(max_depth=2)],
                 learning_rate=[1, 1.5, 2],
                 n_estimators=[100, 200],
                 algorithm=["SAMME", "SAMME.R"],
                 random_state=[2])


parameters_dict = {'Stochastic Gradient Descent' : params_SGD,
                   'Random Forest': params_RF,
                   'AdaBoost': params_AB,
                   'Gaussian Naive Bayes': {},
                   'LogisticRegression': {}
                    }
```

```
#Runs
scoring_metric = 'f1_macro'
model_name, model_parameters, model_best_score = [], [], []
model_names = model_dict.keys()
for MN in model_names:
    print("="*45)
    print(MN)
    clf = model_dict[MN]
    params = parameters_dict[MN]

    gridsearch = GridSearchCV(clf,
                              params,
                              scoring=scoring_metric,
                              cv=10, #Specify number of folds in cross validation
                              verbose=2,
                              n_jobs=1)

    best_model = gridsearch.fit(X_train, y_train)
    model_name.append(MN)
    model_parameters.append(gridsearch.best_params_)
    model_best_score.append(gridsearch.best_score_ )

#Compare different models, get macro-f1 score and optimal parameters chosen for each
model_comparison_df = pd.DataFrame([model_name, model_best_score,
model_parameters]).T
model_comparison_df.columns = ['model_name', scoring_metric,'parameters']

model_comparison_df = model_comparison_df.sort_values(by=scoring_metric,
ascending=False).copy().reset_index()
clf_name = model_comparison_df['model_name'].iloc[0]
```

```
clf_params = model_comparison_df['parameters'].iloc[0]


print('Best model:', clf_name, clf_params)

model_comparison_df


#Model Evaluation

#Testing other classifiers' performances is done by selecting it from the table above

row_number = 0 #Default is 0, as it is the best performing model, but can be changed to view other classifiers' results (0,1,2,3,4)

clf_name = model_comparison_df['model_name'].iloc[row_number]

clf_params = model_comparison_df['parameters'].iloc[row_number]


print('Selected model:', clf_name)


#Model Performance on Training Data

clf = model_dict[clf_name]

clf.set_params(**clf_params)

clf.fit(X_train, y_train)

y_eval_pred = clf.predict(X_eval)


print(classification_report(y_eval, y_eval_pred, digits=3)) #Performance results are rounded to 3 significant digits


#Model Performance on Test Data

clf = model_dict[clf_name]

clf.set_params(**clf_params)

clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)
```

```
print(classification_report(y_test, y_pred, digits=3)) #Performance results are rounded to
3 significant digits


#Confusion matrix
cm = confusion_matrix(y_test, y_pred)


cm_df = pd.DataFrame(cm,
                     index = list(test_labels),
                     columns = list(test_labels))
#Plot the heatmap
plt.figure(figsize=(10, 10))


sns.heatmap(cm_df,
            center=0,
            cmap=sns.diverging_palette(10, 240, s=80, l=55, as_cmap=True),
            annot=True,
            fmt='g')


plt.title(clf_name + '\n unseen F1-score: ' + str(f1_score(y_test, y_pred,
average="macro")), fontsize = 18)
plt.ylabel('True label', fontsize = 16)
plt.xlabel('Predicted label', fontsize = 16)
plt.show()
```

APPENDIX G

BINOMIAL PROBABILITY DISTRIBUTION TABLE

Numbers in the table represent $p(X=x)$ for a binomial distribution with n trials and probability of success p.

Binomial probabilities:

$$\binom{n}{x} p^x (1-p)^{n-x}$$

		p										
n	x	0.1	0.2	0.25	0.3	0.4	0.5	0.6	0.7	0.75	0.8	0.9
12	0	0.282	0.069	0.032	0.014	0.002	0.000	0.000	0.000	0.000	0.000	0.000
	1	0.377	0.206	0.127	0.071	0.017	0.003	0.000	0.000	0.000	0.000	0.000
	2	0.230	0.283	0.232	0.168	0.064	0.016	0.002	0.000	0.000	0.000	0.000
	3	0.085	0.236	0.258	0.240	0.142	0.054	0.012	0.001	0.000	0.000	0.000
	4	0.021	0.133	0.194	0.231	0.213	0.121	0.042	0.008	0.002	0.001	0.000
	5	0.004	0.053	0.103	0.158	0.227	0.193	0.101	0.029	0.011	0.003	0.000
	6	0.000	0.016	0.040	0.079	0.177	0.226	0.177	0.079	0.040	0.016	0.000
	7	0.000	0.003	0.011	0.029	0.101	0.193	0.227	0.158	0.103	0.053	0.004
	8	0.000	0.001	0.002	0.008	0.042	0.121	0.213	0.231	0.194	0.133	0.021
	9	0.000	0.000	0.000	0.001	0.012	0.054	0.142	0.240	0.258	0.236	0.085
	10	0.000	0.000	0.000	0.000	0.002	0.016	0.064	0.168	0.232	0.283	0.230
	11	0.000	0.000	0.000	0.000	0.000	0.003	0.017	0.071	0.127	0.206	0.377
	12	0.000	0.000	0.000	0.000	0.000	0.000	0.002	0.014	0.032	0.069	0.282
13	0	0.254	0.055	0.024	0.010	0.001	0.000	0.000	0.000	0.000	0.000	0.000
	1	0.367	0.179	0.103	0.054	0.011	0.002	0.000	0.000	0.000	0.000	0.000
	2	0.245	0.268	0.206	0.139	0.045	0.010	0.001	0.000	0.000	0.000	0.000
	3	0.100	0.246	0.252	0.218	0.111	0.035	0.006	0.001	0.000	0.000	0.000
	4	0.028	0.154	0.210	0.234	0.184	0.087	0.024	0.003	0.001	0.000	0.000
	5	0.006	0.069	0.126	0.180	0.221	0.157	0.066	0.014	0.005	0.001	0.000
	6	0.001	0.023	0.056	0.103	0.197	0.209	0.131	0.044	0.019	0.006	0.000
	7	0.000	0.006	0.019	0.044	0.131	0.209	0.197	0.103	0.056	0.023	0.001
	8	0.000	0.001	0.005	0.014	0.066	0.157	0.221	0.180	0.126	0.069	0.006
	9	0.000	0.000	0.001	0.003	0.024	0.087	0.184	0.234	0.210	0.154	0.028
	10	0.000	0.000	0.000	0.001	0.006	0.035	0.111	0.218	0.252	0.246	0.100
	11	0.000	0.000	0.000	0.000	0.001	0.010	0.045	0.139	0.206	0.268	0.245
	12	0.000	0.000	0.000	0.000	0.000	0.002	0.011	0.054	0.103	0.179	0.367
	13	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.010	0.024	0.055	0.254
14	0	0.229	0.044	0.018	0.007	0.001	0.000	0.000	0.000	0.000	0.000	0.000
	1	0.356	0.154	0.083	0.041	0.007	0.001	0.000	0.000	0.000	0.000	0.000
	2	0.257	0.250	0.180	0.113	0.032	0.006	0.001	0.000	0.000	0.000	0.000
	3	0.114	0.250	0.240	0.194	0.085	0.022	0.003	0.000	0.000	0.000	0.000
	4	0.035	0.172	0.220	0.229	0.155	0.061	0.014	0.001	0.000	0.000	0.000
	5	0.008	0.086	0.147	0.196	0.207	0.122	0.041	0.007	0.002	0.000	0.000
	6	0.001	0.032	0.073	0.126	0.207	0.183	0.092	0.023	0.008	0.002	0.000
	7	0.000	0.009	0.028	0.062	0.157	0.209	0.157	0.062	0.028	0.009	0.000
	8	0.000	0.002	0.008	0.023	0.092	0.183	0.207	0.126	0.073	0.032	0.001
	9	0.000	0.000	0.002	0.007	0.041	0.122	0.207	0.196	0.147	0.086	0.008
	10	0.000	0.000	0.000	0.001	0.014	0.061	0.155	0.229	0.220	0.172	0.035
	11	0.000	0.000	0.000	0.000	0.003	0.022	0.085	0.194	0.240	0.250	0.114
	12	0.000	0.000	0.000	0.000	0.001	0.006	0.032	0.113	0.180	0.250	0.257
	13	0.000	0.000	0.000	0.000	0.000	0.001	0.007	0.041	0.083	0.154	0.356
	14	0.000	0.000	0.000	0.000	0.000	0.000	0.001	0.007	0.018	0.044	0.229

(continued)