5-2020

Toward More Predictive Models by Leveraging Multimodal Data

Sudarshan Srinivasan
University of Tennessee, ssriniv3@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Sudarshan Srinivasan entitled "Toward More Predictive Models by Leveraging Multimodal Data." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Science.

Gregory Peterson, Major Professor

We have read this dissertation and recommend its acceptance:

Edmon Begoli, Amir Sadovnik, Anahita Khojandi

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

TOWARD MORE PREDICTIVE MODELS BY LEVERAGING MULTIMODAL DATA

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Sudarshan Srinivasan

May 2020

DEDICATION

I dedicate this dissertation to my family who have been with me throughout the arduous journey of my Ph.D.

ACKNOWLEDGMENT

There are many people who have been of immense help during my Ph.D. I want to particularly thank 3 people without whom I would not have made it this far. First and foremost, I would like to thank my advisor Dr. Gregory Peterson who gave me invaluable advice on my academic and Ph.D. life, during countless hours of conversations, which inspired numerous ideas that evolved into my dissertation. I also thank Dr. Peterson for his indispensable advice on scientific writing and presentation.

Next, I would like to thank my supervisor at Oak Ridge National Lab (ORNL) Dr. Edmon Begoli. He funded my research and provided crucial advice on time and project management. Without him, this research would not have been possible and for that, I'm immensely grateful.

I would like to thank my research collaborator from Emory University (formerly at the University of Tennessee Health Science Center) Dr. Rishikesan Kamaleswaran who helped me to design my experiments in imminent ICU admission prediction which forms a core part of this research. I'm very grateful for his clinical informatics expertise that helped a novice like me navigate the intricacies of healthcare.

I would like to thank my friends Kris Brown, Eduardo Ponce, and Corey Johnson for all their help and emotional support through this journey. Finally, I thank my family for providing the strength and support I needed to pursue and complete this research.

ABSTRACT

Data is often composed of structured and unstructured data. Both forms of data have information that can be leveraged by machine learning models to increase their prediction performance on a task. However, integrating the features from both these data forms is a hard and complicated task. This an even more true for models which operate under time-constraints. Time-constrained models are machine learning models that work on input where time causality has to be maintained, such as predicting something in the future based on past data. Most previous work does not have a dedicated pipeline that is generalizable to different tasks and domains, especially under time constraints.

In this work, we present a systematic, domain-agnostic pipeline for integrating features from structured and unstructured data while maintaining time causality for building models. We focus on the healthcare and consumer market domain and perform experiments, preprocess data, and build models to demonstrate the generalizability of the pipeline. In particular, we focus on identifying patients who are at risk of imminent ICU admission. We use our pipeline to solve this task and show how combining structured data and unstructured data machine learning improves model performance by up to 8.5%.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Data is the raw material for building machine learning (ML) models for solving different tasks [1]–[5]. The data can belong to any domain ranging from consumer businesses, to stock markets, to healthcare. The tasks themselves, while different, are often tied to the specific domain.

Data can predominantly be categorized into structured and unstructured data [6]. Structured data includes information with a high degree of organization. Examples of structured data include a product category, values of vital signs, or clinical lab tests. In contrast, unstructured data is unorganized and does not hew to conventional data models. Free-text such as the description of a product or a clinical note written by a doctor are examples of unstructured data. Finally, data can also have a time component where the meanings of data recorded in the past are different than those recorded in the future.

Integrating heterogeneous data while preserving information and time causality from different sources is complicated. Despite this complexity, one could contemplate that building ML models by combining the information contained in both forms of data, would boost performance and result in a more robust ML model, as these models have access to additional information. In particular, we are addressing the problem of developing a ML pipeline for incorporating temporal constraints across structured and unstructured data.

While there are many business projects and Kaggle competition solutions that use different data sources to build their ML models, there is a lack of a systematic pipeline that goes from raw data to a finished task that is *generalizable* across different domains.

This is especially true in the medical domain where there is an abundance of all forms of data in the form of electronic health records (EHR) which include both structured data in the form of diagnosis codes, vital signs, and lab tests and

unstructured data in the form of clinical notes written by healthcare providers during or after patient encounters [7]–[10].

While data is abundant in one field for one task, it is also possible for data to be sparse in another field or another task. Data sparsity results in models being underfit and not generalizable. A popular attempt to solve this problem is using *transfer learning*. Transfer learning is a research probem in machine learning where the knowledge gained from solving one problem is transferred and applied to another problem in a different but related domain [11]–[14]. In transfer learning models are trained on a particular task or data and these trained models are reused directly or by training them further on a target task or data. This is based on the assumption that the source and target are related in some form. This technique is especially useful when the data distribution of the source and target domains are drastically different in terms of positive class representation.

In this work, we build a systematic pipeline for integrating features from both structured and unstructured data while maintaining time causality to train and evaluate ML models for different tasks. We show that this pipeline is generalizable to different domains by applying it to different tasks from the medical field and the e-commerce field. Furthermore, we define *cross training* which is a type of transfer learning where we train ML models on a source dataset, further train the models on a portion of the target dataset, and then test them on the remaining portion of the target dataset.

We define the important medical task of predicting imminent intensive care unit (ICU) admission and build models to accomplish this task. Predicting imminent ICU admissions have important implications in critical care medicine. We do this by using two different datasets and using structured data such as vital signs and lab tests along with unstructured data consisting of clinical notes written by healthcare professionals. We show that by augmenting the clinical notes with structured data, we can get comparable or better performance and more importantly gain insights into the modeling process. Finally, we perform transfer learning between the two datasets and show we can get increased performance on a very imbalanced dataset by transferring knowledge from a model trained in a different dataset.

In the same vein, we use a consumer dataset from a shopping chain to predict the listing price of an item based on the structured information about the item

along with the unstructured free-text description of the item. We use the same ML pipeline for both the tasks to test the generalizability of the pipeline and its application to different domains.

Our contribution in this work is developing a robust ML pipeline for incorporating time constraints across structured and unstructured data. We demonstrate this pipeline by predicting imminent ICU admission that uses both structured and unstructured data. We demonstrate the generalizability of the pipeline by performing a task from the e-commerce fields. Finally, we demonstrate a transfer learning approach to help increase performance in imbalanced datasets.

This dissertation is organized as follows: In chapter 2 we present background work and related literature. In chapter 3 we talk about the various methodologies that we use in this work along with various evaluation metrics. We also define several algorithms which are used for building and evaluating our models. In chapter 4, we show the potential of building machine learning models using multimodal data by applying our pipeline to a consumer market task of predicting the price of an item given its characteristics as structured information and user-defined item description as unstructured data. We then move on to the healthcare domain in chapter 5, where we define the important task of predicting imminent ICU admission and describe our novel way of integrating time-based structured and unstructured data. In chapter 6, we demonstrate the power of transfer learning as part of on-going work where we train our models on a source dataset and further train them on a portion of the target dataset and test them on the remaining portion. Finally, we conclude our work in chapter 7 and describe ideas for future work in this research.

CHAPTER 2
LITERATURE OVERVIEW

In this chapter, we provide a literature overview of recent work pertaining to our research. We review work that uses structured data and unstructured data, how it has been integrated in the past, and how that is different from our approach. We also review work on transfer learning. Finally, we review previous work from the healthcare domain specifically tied to our task of imminent ICU admission prediction.

While the specific type of data we are looking at exists in different domains, there is a lack of published research in non-healthcare domains. Much of the work in non-healthcare domains such as retail, sales, marketing, and business can be found as part of Kaggle [15] competition solutions where competitors voluntarily publish their work in the form of code and explanations. This is true especially for data that contains a mixture of structured and unstructured data as this type of data is scarce in non-healthcare domains.

2.1 Structured Data

Structured data use in machine learning is ubiquitous. Most industry have some form of structured data associated with their domain and use them with machine learning to build predictive models. Furthermore, we can find a lot of work in terms of explanation blogs and code from Kaggle that fit the structured data modality. As such, the amount of work that use machine learning on structured data is vast and reviewing them would be out of the scope of this dissertation. For a good review of use of machine learning algorithms on structured data please refer to [16]–[19]. Our main focus for this section of the literature review will be the use of structured data in the healthcare domain.

This is because, there is a lack of published work that uses only structured data outside of the healthcare domain. As mentioned earlier,

Electronic health records (EHR) data readily falls into the format of structured and unstructured data and there has been a lot of published work in this area indicating a higher impact of our research in the healthcare domain. Hospitals generate large volumes of data which include discharge summaries, records of medicines administered, laboratory results, and treatments provided which are stored in the form of EHRs [20]. There has been a lot of interest in applying data analytics in healthcare [21]–[23] and EHR data mining [24]–[26]. With the success of deep learning [27] techniques in other domains, there has been a tremendous effort to bring it to healthcare [7], [28]. State-of-the-art techniques have been applied to get great results to pieces of EHR for various tasks. However, only a hand full of work has been done in utilizing both structured (e.g., diagnosis codes) data and unstructured (e.g., clinical notes) data together for target tasks.

In [29], only structured data in the form of continuous variables were used to diagnose patients. Diagnosis is done by predicting diagnostic codes assigned to each patient during their visit given their history of 13 continuous variables during each episode for all patients. The input data is considered as a clinical time series and the problem of diagnosis is formulated as a multi-label classification problem. They use Long Short-Term Memory (LSTM) networks, a variant of recurrent neural networks (RNN), for modeling the frequently but irregularly sampled time series of the input variables. They evaluate their system's performance by comparing it to other techniques for predicting diagnostic codes. With a similar objective of predicting clinical codes, in [30], the authors develop *DoctorAI* which uses another variant of RNN called the Gated Recurrent Unit (GRU). Each patient's visit is time-stamped with clinical codes and these are the inputs to their system along with the duration since the last visit. The system is trained in a single supervised learning scheme where the outputs are possible clinical codes for the next visit and the time to next visit. In particular, the clinical codes for each patient during each visit are embedded into a lower-dimensional space from both a multi-hot encoding and by using a matrix generated by the skip-gram algorithm. The weights from the embedding layer are fed into a recurrent layer with a GRU and finally, there are two independent layers, one with a softmax with a multi-label output to predict the clinical code and rectified linear unit (ReLU) for predicting the time to next visit.

In [31], clinical codes from EHR data are represented by embedding them into a low-dimensional vector space. Their system *eNRBM* uses modified Restricted

Boltzmann Machines (RBM) to model EHR using the demographics, admission, diagnosis, and intervention codes as input. Their representation is evaluated on a suicide risk assessment task based on a 48-month history. The history is split into disjoint intervals of predefined lengths, where each interval contains time-stamped variables. These input variables are fed into a "representation layer" whose dimensions are less than the number of input variables, and the output of this layer serves as the representation of the input variables, which is then fed as input to the target task.

In [32], a deep learning framework called *DeepCare* is presented which reads medical records consisting of clinical codes and uses a modified LSTM network for predicting future medical outcomes. In particular, diagnosis and intervention codes are embedded into a vector space and these embeddings are learned as part of the training process. The input to the system is a sequence of admissions, where each admission contains embeddings for the codes, admission method (planned or unplanned), and the time elapsed since the previous admission. They modify the standard LSTM to handle irregular timings found in EHRs and enhancing the effect of admission and intervention type. They evaluate their system by predicting the risk of unplanned readmission of patients suffering from diabetes.

In [33], a deep neural network for predicting the mortality of a patient within 3-12 months from a prediction date is presented. The objective is to automatically identify patients who are likely to benefit from palliative care services and short-term mortality prediction is considered as a proxy for the target task. An observation window of 12 months before the prediction date is divided into 4 unequal slices and clinical code counts for each window along with summary statistics of the counts are taken and concatenated together to form a feature vector. These are then fed as input to deep neural networks for a supervised binary classification problem of mortality prediction. Furthermore, to facilitate the interpretation of the decision made by the model, they develop a technique wherein they ablate each code category and run the model to the modified input to note the probability drop of the prediction. They reason that those codes which were ablated that resulting in a high drop in probability has a higher influence in the model's decision for that patient.

The literature we reviewed so far uses structured data (or a subset of it) of the EHR and discards the information hidden in the clinical text. The difference

between our approach and the ones we have seen above is we do not use deep learning approaches and we also combine the structured data with unstructured textual data. Our approach is more systematic and can potentially be applied to other domains as an algorithm.

2.2 Unstructured Data

Unstructured data mainly contains free-text and is often highly heterogeneous. This is especially true for clinical notes which are rich in author and domain-specific idiosyncrasies, abundant abbreviations, rich medical jargon, and spelling and typing errors [24].

For any task involving text, representation of that text in a suitable format that can be fed into a ML model is required. One of the ways to represent clinical texts is to represent clinical concepts such as clinical codes embedded in the texts. These concepts have unique identifiers called CUIs. In [34], the free text is converted to concept sequences and embeddings for the concept unique identifiers (CUI) are learned using *word2vec*'s skip-gram model [35], corresponding to predicting nearby CUIs from each other in the given context of medical journals. In [36], clinical texts from insurance claims and publicly available EHRs are used to create concept embeddings. After the clinical concepts are extracted, they are preprocessed into a form suitable to be fed into *word2vec*. In particular, they time-stamp each concept and combine each time-stamped concept into a "sentence" to create embeddings for the concept. Similar to this work is [37], where they present a framework called *cui2vec* to create concept embeddings from a combination of clinical notes from EHRs, information from an insurance claims database, and text from biomedical journal articles. Unlike previous works, here once the CUIs are extracted a CUI-CUI co-occurrence matrix is constructed for each data source. Once the master co-occurrence matrix is created, it is directly used to create *GloVe* [38] or *word2vec* style embeddings. These pre-trained CUI embeddings are publicly available.

Topic models [39] are used in [40], again for patient mortality prediction with notes from the MIMIC-II [41] dataset. They use Latent Dirichlet Allocation (LDA) [42] to create the topics from the notes and then build a linear support vector machine (SVM) [43] model to predict mortality outcomes. Unlike previous work, where the discharge summary was discarded from the input to prevent data leakage, here all the notes were included. A non-parametric topic modeling

technique known as the Hierarchical Dirichlet Process (HDP) is applied for the same task using only the nursing notes from the MIMIC-II dataset in [44]. For each patient, a "document" was created representing a collection of UMLS code, representing either a disease, symptom, medication, or procedure. Once the topics were obtained, multivariate logistic regression was used to find the association between each topic and hospital mortality where the proportion of words assigned to each topic was used as input [45]. Finally, in [46], the authors present *TopicRNN* that combines latent topic models and RNN based language models for capturing global semantic meaning relating words in a document.

In [47], convolutional neural networks (CNN) based neural document embeddings are used for patient mortality prediction. Word embeddings of sentences from clinical notes in the MIMIC-III [48] dataset are fed into a 2-layer CNN, whose first layer maps the sentences to sentence vectors and second layer combines the sentence vectors into a single patient representation. In addition, they map the 14 types of note categories into vector space and concatenate the note type vector with the input word embedding vector. By doing this, the authors claim that this information access how important are the individual sentences to the target task. Similarly, in [49] LSTM networks are combined with topic models to predict mortality. Here the notes are represented by a bag-of-words model instead of word embeddings. For each patient, the time elapsed since admission in 12 hour-long segments is defined as *time points*, and clinical notes within a time point are aggregated into one document. The topics are represented in a layer in the LSTM network and are either trained using LDA or using encoders and decoders. The authors acknowledge topic interpretability and quality is bad with the top words in topics being rare words or medical jargon.

In this section we reviewed literature that use unstructured free-text as the single source of data into their ML models, discarding the associated structured data. Furthermore, some of these work use advanced representation techniques for the free-text which tend to take too long to train. In our work, we use standard NLP techniques for processing text, while also affording a way to swap in more advanced methods via our generalized pipeline and framework. Our framework also provides a way to integrate structured data to the features extracted from unstructured data.

2.3 Multimodal Data

The idea behind combining structured and unstructured data is to use the information from both to inform the final model that is being trained on an application. While there are different ways to do this, the inherent complexity of combining different data sources makes this task difficult, even more so in the case of EHRs. Here we present some of the work that has been done in this realm.

In [50], the authors built a manual pipeline to identify cohorts of ICU patients who received dialysis. Basic information retrieval tools are used to extract information from both forms of data. In particular, SQL queries are used on the MIMIC-II dataset to search the database for codes and terms codes for information associated with ICU dialysis. These were then manually reviewed without any machine learning procedure involved. Their main conclusion is that using data from both sources gives maximum performance. In [51], the authors manually extract features from discharge summaries and augmented with structured information extracted from EHRs to identify whether any complications occurred during the length of stay of the patient. Here a machine learning model was built using the manually extracted features. From structured data, they extract information related to medications, clinical events, culture reports, and radiology reports. Clinical texts were preprocessed to extract disease mentions, negations, uncertainty modifiers, or correlated words and phrases. All these features were then fed into a binary logistic regression model to classify the length of stay. They compare the relative performance of models with both types of input, i.e., only clinical text features and both text and extracted features. Their results indicate that using features from both forms of data increased performance.

Scheurwegs et al. combine structured and unstructured textual data to assign clinical diagnostic and procedural codes to patient stays [52]. Structured data included lab results, inpatient medication prescriptions, pathology, procedure, and medication codes. Categorical variables are represented by counting the number of occurrences of distinct assigned codes. Also, another set of features from the structured data consists of multiple "meta-features" such as the average amount of assigned codes per day and the total amount of unique codes per stay. Unstructured data included letters, notes, protocols, and attestations. These were processed using a combined bag-of-words (BOW) of all documents of a certain type associated with a specific stay. The data is combined using either early data

which consists of integrating the features of different sources before training the model or late data integration where prediction results from separate models, trained on each distinct source, are used as input for a second classifier. They do a comparison study and find that late integration with both forms of data performs best in assigning codes to patient stays. While this work is similar in objective to ours, their methodology uses traditional ML techniques and feature engineering. Furthermore, they do not provide a united representation of the data.

In [53], Bai et al. learn a joint representation of medical concepts using diagnostic codes from structured data and extracted words from clinical notes. They introduce a *JointSkip-gram* model, which is a variant of the skip-gram model from word2vec [35], where the diagnostic codes and word sequences from clinical notes are used to predict neighboring codes and sequences. This way they are able to learn a relationship between the medical codes and words in clinical notes. They evaluate their algorithm using the MIMIC-III dataset by training vector representations of the codes and words and testing them on a task of predicting diagnosis on a future visit. Unlike our work, they do not utilize the real valued information from the structured data and they consider the diagnostic codes as a "bag" of codes thereby losing the information about when these were recorded.

In [54], a deep learning framework called *DeepPatient* where each patient is described by a single vector or by a sequence of vectors computed in predefined temporal windows. In particular, from the structured data demographic details (such as age, race, and gender), and common clinical descriptors (such as medical codes) were taken along with the free-text clinical notes. The structured data was processed by simply counting the presence of medical codes. The clinical notes were preprocessed to remove identified negated tags, flagging and differentiating family-related and patient-related tags. The authors use topic modeling [55] to represent each note with a vector of 300 topic probabilities. All these feature values are then normalized and fed into a 3-layer stacked denoising autoencoder (SDA), where the middle layer once trained would hold the representation of each patient. They evaluate the performance of their system by predicting the probability that a patient might develop a certain disease in the future given their current clinical status. They build a random forest classifier by feeding it as input, their deep representation along with raw EHR and alternate representations (such as PCA) and show that their representation gives better predictions.

In [56], information from both data is combined to predict health outcomes for patients with Congestive Heart Failure (CHF) using the MIMIC-II dataset [41]. In particular, the structured data is embedded into a 10-dimensional vector space with the embeddings being learned as part of the training process. The clinical notes are represented using *word2vec* which was pre-trained on Wikipedia articles with a 150-dimensional semantic representation. In addition, waveform data from the electrocardiogram (ECG) signal for patients are computed using standard time-series analysis and are also included as part of the input. All these components are then fed into a hidden layer which is then passed to a softmax layer that predicts the health outcome of the patient. Health outcome is determined as improving if the patient is discharged from the ICU with clinician approval and as not improving if the patient is transferred to specialized facilities or if the patient dies.

Rajkomar et al. propose a method to represent a patient's entire raw EHR record in a single data structure that can then be used for predictive tasks [57]. The representation is based on the Fast Healthcare Interoperability Resources (FHIR) [58] format. In their work, they take EHR data as input and produce FHIR outputs by mapping various FHIR resources including time (by using the difference to the time of prediction) to tokens and representing them by embeddings. Both the structured data and the clinical notes are represented by unique "tokens" which are obtained by preprocessing the data. These tokens are then fed into an embedding layer to create embeddings. These embeddings are fed into an LSTM recurrent neural network for various downstream applications and are learned during training. In particular, the authors devise a single data structure to make predictions for mortality, 30-day readmission, length of stay, and a full diagnosis of the patient.

In [59], the authors propose a general multi-task framework for forecasting the onset of diseases prior to diagnosis time using both clinical notes and structured data. They focus on CHF, kidney failure, and stroke with the raw text of clinical notes, lab, and vital sign data recovered from them, and structured demographics data serving as input. Different deep learning architectures are built and compared against each other to assess the contribution of the information in the clinical notes to the overall performance. They consider clinical notes over a period of 3 years with training data over a year with a 3-month gap after which they predict disease

over a 6-month window. In addition to training new word embeddings on their data initialized with pre-trained embeddings, they also detect negation and replace negated words with the negative of the corresponding word embedding. They build 3 models: a CNN, a bidirectional LSTM (BiLSTM), and a hierarchical model that uses a CNN and a LSTM together. Their results show that the BiLSTM model with features including demographics, lab, and vital sign data, clinical notes, and negated word embeddings outperforms all other models.

In [60], the authors tackle the problem of predicting ICD-10 diagnostic codes by using multimodal data. In particular, they build individual models trained on different forms of data including unstructured text, semi-structured text, and structured tabular data, and use an average model ensembling strategy to predict the ICD-10 codes based on the input. They use various forms of CNN for building models on unstructured text and semi-structured text and use a classical decision tree on the structured data. They report that their best performing ensemble model significantly outperforms the baseline models which only use textual data. While this work is similar to ours, a key difference is in the way the multimodal data is leveraged. In our work, we integrate both structured and unstructured data before feeding it to single ML model. This strategy affords us to get a unique representation of the entire data as opposed to building individual models for different types of data. Furthermore, their approach is not directly applicable to time-based data as different data points might be taken at different time intervals. This might enable models trained on certain data to have access to information that other models might not have.

As we have seen in this section there has been several approaches in integrating structured data and unstructured data. However, these approaches lack in two key areas: 1) they are task-specific and domain-specific; 2) they are not directly applicable to time-based data which are recorded at varying frequencies and time intervals. In this work, we present a novel way of combining time-based data and present a generalized pipeline that can potentially be applied to different tasks across different domains.

2.4 Transfer Learning using Healthcare Text

Transfer learning is used to improve a model in one domain by transferring information from a related domain [12], [13]. Typically, the source domain contains

easily obtained data, while the target domain contains insufficient data. This is especially useful for clinical notes, which often have access restrictions and require domain expertise to label the notes properly.

Transfer learning in NLP has gained traction in recent years. Subsequently, its use in healthcare was just a matter of time. In [61], the effectiveness of transfer learning using contextualized embeddings for clinical concept extraction is explored. In particular, the authors compare the use of word2vec, GloVe, and fasttext [62], [63] embeddings against the more advanced embedding methods and representations provided by ELMo [64] and BERT [65]. They hypothesize that certain types of diseases or conditions normally appear in specific contexts with similar grammar structures, emphasizing the need to take advantage of contextualized entity representations. Furthermore, they compare the effects of pre-training, a form of transfer learning, on a large amount of open-domain data versus clinical notes from MIMIC-III. They compare different embeddings based on the performance of the end model's target task, which is a sequence labeling bidirectional LSTM with a final conditional random field (CRF) layer (Bi-LSTM CRF). Their results show that the best model performance was achieved using BERT pre-trained with clinical domain embeddings. This emphasizes the fact that, for certain focused task, contextualized embeddings outperform traditional embeddings and large improvements can be achieved by pre-training a model from a large corpus, followed by task-specific fine-tuning.

In [66], researchers analyze the effectiveness of using transfer learning to conduct efficient patient-level clinical note medical concept representations. They achieve this by using these representations for predicting future clinical events of interest, such as mortality, inpatient admissions, and emergency room visits. The aim is to use data-driven methods to reduce the need for extensive feature engineering by automatically learning how to summarize a sequence of clinical notes about a given patient. For this purpose, two approaches are explored. One entails using GloVe to learn embeddings for biomedical concepts mentioned in the clinical text. These embeddings are then aggregated to form a representation of a single note for a particular patient, obtained by further aggregating representations of these notes. The other approach uses RNNs to simultaneously learn representations of concepts, notes, and patients, achieved by training the model on a proxy source task of predicting diagnosis codes for each patient. GloVe and RNNs create embeddings

from biomedical terms, which are then used as input features to a logistic regression model for the target tasks. Along with concept embeddings, the authors build simple BOW models with Term Frequency-Inverse Document Frequency (TF-IDF) representations. They compare the model performance across using embeddings that were learned with differing amounts of training set size. They show that transfer learning can be used as an effective way to engineer task-specific features when training data is limited.

In [67] the authors explore the use of transfer learning to automatically assign ICD-9 codes to discharge summaries of patients from MIMIC-III by first training a deep learning model on a much larger but related dataset, the BioASQ [68]. The objective here is to assign relevant MeSH terms based on a manual reading of scholarly publications by human indexers. The parameters of this trained model are then transferred and fine-tuned for predicting the ICD-9 codes on the target dataset. The authors argue that the discharge summaries in MIMIC-III database have the following drawbacks: 1) the distribution of ICD-9 code is highly biased; 2) there is a large variation in the cardinality of each code; 3) the average number of discharge summaries per code is small; 4) and finally, there is a large variation in the length of the discharge summary. Conversely, the MeSH data from the BioASQ3 dataset has more samples and labels and the average samples per label are much larger. They hypothesize that training a model on this large dataset will capture features relevant to assigning codes to medical terms and that transferring this knowledge will improve performance. They build multi-scale CNNs where the input is fed to different convolutional kernels at the same time and the outputs of these kernels are concatenated in the next layer. This is an efficient method to combine different features for classification and to obtain multiple local contextual feature maps. The input is the word embeddings of the medical free text and the output is a sigmoid activation function to predict the probability of a label. To test the efficacy of multi-scale CNN, they also build a sequential CNN model for the transfer learning task. Additionally, to see how the number of training samples impacted the effectiveness of transfer learning, they build multiple multi-scale CNN models that use a varying amount of samples from the source dataset. Their results indicate that the multi-scale CNN with transfer learning built with 100% of the samples from the source dataset outperformed all other models.

Transfer learning is a powerful approach to train ML models in scenarios where there is a lack of sufficient data. As we have seen, this is especially true for healthcare domains, in which privacy concerns restrict the amount of data available for analysis. Following in the footsteps of the previous work in transfer learning, we implement transfer learning between different datasets for the same task. While we currently only use only clinical notes in this work due to the lack of access to the entire data, our framework can easily be extended to include structured data, which could potentially lead to better model performance.

CHAPTER 3
METHODOLOGY & IMPLEMENTATION

Building machine learning models involves preprocessing the data, applying the appropriate machine learning algorithm, and interpreting the results. There different methods of preprocessing data, different types of data, and different algorithms that we can try. In this chapter, we highlight some of the preprocessing techniques and machine learning algorithms that we used in this work.

3.1 Preprocessing Techniques for Text

When it comes to natural language processing (NLP), there are two standard steps of preprocessing to prepare the data for modeling. The first step is tokenization, where a string of words is segmented by dividing the string into its component words called *tokens*. The next step is vectorization, where these tokens are used to represent the document into vectors of numbers.

3.1.1 Tokenization

Tokenization is the process where a string of words is segmented into its component words. This is an important step for preparing the text to be fed into machine learning algorithms. There are many ways to tokenize text with varying degrees of complexity.

One of the most basic ways to tokenize text is to just split the words based on whitespace. This method is incredibly simple which is both its advantage and disadvantage. Another disadvantage is that similar words will appear as distinct tokens. For example, the last word in a sentence is not normally separated by a space from the period at the end of the sentence. By splitting on whitespace, the word along with the period is considered as a distinct token than the word itself which can lead to increased vocabulary size. Splitting on whitespace and considering each word as a unique token is known as *unigram* tokenization. A more general method is the *n-gram* tokenization, where n words are grouped together

and split by whitespace. Using n-gram tokenization, it is possible to capture n-word phrases as unique tokens.

3.1.2 Vectorization

Vectorization involves converting tokens into numbers which can then be processed by machine learning algorithms. Similar to tokenization schemes, there are many ways to vectorize a token that vary in complexity. One of the most basic ways of vectorizing a corpus of tokens is to replace each token with its count of occurrence in the document. This is known as *count vectorization*. While simple, it has a disadvantage of creating sparse representations and giving equal importance to all words in the corpus ignoring their relative importance.

Term Frequency Inverse Document Frequency (TF-IDF) is a "numerical statistic that is intended to reflect how important a word is in a corpus"[1]. It is composed of two components the *term frequency* (TF) and the inverse document frequency (IDF) and the final score is a product of these two values. The TF of a token depends on the number of occurrences of the term in a document, while the IDF depends on the number of documents that contain that token. More information about TF-IDF weighting can be found in [69]. Each document is then represented by a vector of the TF-IDF scores of each token in the document. Although this method also produces sparse representations, the values of the TF-IDF take into account the relative importance of each word.

3.2 Machine Learning Algorithms

In order to effectively gauge the strengths and weaknesses of the models that we build, we need to compare them against different baselines. We built multiple models for the tasks using different machine learning algorithms which we highlight here.

3.2.1 Logistic Regression

Logistic regression (LR) is one of the most basic and simplest modeling technique that is used to model the probability of a certain event or class. LR is a statistical model that in its basic form uses a sigmoid function to model a binary dependent variable. A sample sigmoid function is shown in figure 3-1.

[1] https://en.wikipedia.org/wiki/Tf-idf

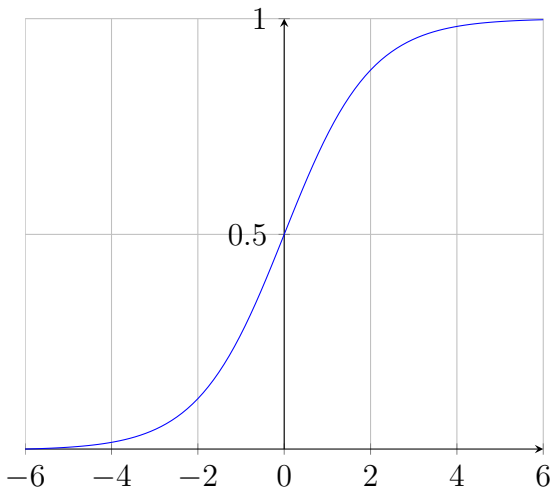


Figure 3-1. Sigmoid Function

LR is a linear model in that it does not include any nonlinear components and is a linear combination of one or more independent variables. Until very recently, LR has been very widely used in medical applications due to its simplicity and the use of model coefficients can be interpreted as odds ratios that are clinically meaningful [70]. However, this simplicity also serves as a disadvantage as LR being a generalized linear model cannot capture complex and nonlinear relationships in the data. For our implementation of LR, we use the Scikit-Learn library [71].

3.2.2 Random Forests

Random Forests (RF) [72] are an ensemble learning method used for classification, regression, and other tasks. They are trained by constructing multiple decision trees and during inference they output the class that receives the maximum vote in case of a classification problem or the mean prediction of the outputs in case of a regression problem. They combine multiple decision tree predictions, such that each tree depends on the values of a random vector sampled independently and with the same distribution.

RF's have several advantages. First, it uses a technique known as *bagging* [73], which is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. This tackles overfitting by reducing the variance of the predictors. They do not require any feature scaling and handle nonlinear parameters efficiently. RF's are robust to outliers and are very fast to train. Their

disadvantage includes their complexity and longer prediction time. Furthermore, more accurate ensembles require more trees, which means using the model becomes slower. For our implementation of RF, we use the Scikit-Learn library [71].

3.2.3 Gradient Boosting Machines

Gradient boosting machines (GBM) [74] are another type of tree-based algorithm. GBM uses a technique known as *boosting*, in which a set of weak learners creates a single strong learner. Due to boosting, GBM does not overfit the data. Furthermore, while GBM takes slightly longer to train, they work much faster than RF during test time. The longer training time is due to the fact that GBMs use stochastic gradient descent to train. Unlike RF, GBM trees are trained in sequence. Furthermore, GBM's are harder to train due to the requirement of careful hyperparameter tuning. In our implementation for GBM we used LightGBM [75].

3.2.4 Ensembling

Ensembling is the process of creating multiple models and combining them to produce an output. More specifically, *ensemble learning* is an umbrella term for methods that combine multiple learners to make a decision, typically in supervised machine learning tasks [76]. In particular, multiple models are trained separately using different algorithms and their outputs on the test set are combined using an ensemble strategy. This is distinct from ensembling techniques such as bagging where ensembling is done during training and use the same algorithm. Simple ensembling involves either taking the average of results output by the different algorithms or using a maximum vote.

3.2.5 Deep Learning

As we saw earlier in chapter 2, deep learning (DL) techniques are widely being used in various ML problems and have shown success in different areas such as computer vision and natural language processing. Deep learning models are based on artificial neural networks and can have highly complex architectures that help them accomplishing highly complex tasks with great performance. However, DL models require lots of data and compute power. In our work, we built standard feedforward neural network models for our tasks. However, given the amount of training time taken to build the models, they yielded results that are comparable to

those models that had a shorter running time. We, therefore, did not experiment with them further and have not included those results as part of this work. We speculate that this low performance could be due to the scarcity of the our data, in particular, our unstructured data. It is important to note that, despite the fact we have not used DL models, our generalized pipeline can be used with any ML model.

3.3 Model Building and Evaluation

In our work, we will be building multiple models using different algorithms for the same task. While the *no free lunch* (NLF) theorem [77] states that "any two optimization algorithms are equivalent when their performance is averaged across all possible problems", we can still compare the performances of the models tuned to the task and data. Algorithms 1 and 2 show how we build and evaluate our models respectively.

We first split the data into a training and testing set, perform training, and then select optimal threshold and optimum hyperparameters in the model building stage.

We also perform model ensembling using two ensembling strategies: average and maximum. Algorithm 3 shows this ensembling algorithm. The ensembling algorithm is run on multiple test sets (same number as the one in algorithm 2) and the results of each individual model is aggregated based on the ensembling function.

Algorithm 1: Model Building Algorithm

Data: Dataset $\mathcal{S} = \{X, \mathcal{P}'\}$ and model $\mathcal{M}$
Result: Model hyperparameters $\mathcal{H}$ and discrimination threshold δ
1 Split the dataset $\mathcal{S}$ into training data $\mathcal{S}_{train}$ and testing data $\mathcal{S}_{test}$ with a 85%-15% split using a fixed seed
2 Train model using examples from $\mathcal{S}_{train}$
3 Get probabilities of the positive class $\mathbb{P}$ using examples from $\mathcal{S}_{test}$
4 Select optimal discrimination threshold δ
5 Get predictions $\mathcal{P}$ by comparing probabilities $\mathbb{P}$ with discrimination threshold δ
6 Compare predictions $\mathcal{P}$ with ground truth $\mathcal{P}'$ to compute performance metrics $\mathcal{T}$
7 Perform hyperparameter optimization to get the optimal hypterparameters $\mathcal{H}$

Algorithm 2: Model Evaluation Algorithm

Data: Dataset $\mathcal{S} = \{X, \mathcal{P}'\}$, model $\mathcal{M}$, hyperparameters $\mathcal{H}$, discrimination threshold δ, start seed *seed*, and number of trials n

Result: Mean performance metrics $\mathcal{T}$

for $s \leftarrow seed$ **to** $seed + n$ **do**
  Split the dataset $\mathcal{S}$ into training data $\mathcal{S}_{train}$ and testing data $\mathcal{S}_{test}$ with a 85%-15% split using seed s
  Initialize model $\mathcal{M}$ with hyperparameters $\mathcal{H}$
  Train model using examples from $\mathcal{S}_{train}$
  Get probabilities of the positive class $\mathbb{P}$ using examples from $\mathcal{S}_{test}$
  Get predictions $\mathcal{P}$ by comparing probabilities $\mathbb{P}$ with discrimination threshold δ
  Compare predictions $\mathcal{P}$ with ground truth $\mathcal{P}'$ to compute performance metrics $\mathcal{T}_s$
end
Compute mean performance metrics $\mathcal{T} = \frac{\sum_{s=1}^{n} \mathcal{T}_s}{n}$

Algorithm 3: Ensemble Algorithm

Data: Test Dataset $\mathcal{S}_{test} = \{X, \mathcal{P}'\}$, model array $\mathcal{M}_E = [\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n]$, discrimination threshold array $\delta_E = [\delta_1, \delta_2, \ldots, \delta_n]$, ensembling function $\mathcal{F}$

Result: Ensemble prediction $\mathcal{P}_E$

Initialize empty array $\mathcal{P}_E$
for *all subsets* $\mathcal{M}_s$ *and* δ_s *of* $\mathcal{M}_E$ *and* δ_E *with* $length(\mathcal{M}_s) > 1$ **do**
  Compute ensemble threshold $\delta_e = \mathcal{F}(\delta_s)$
  Get probabilities of the positive class $\mathbb{P}_s$ for each model in $\mathcal{M}_s$ in test set $\mathcal{S}_{test}$
  Compute ensemble probabilities $\mathbb{P}_e = \mathcal{F}(\mathbb{P}_s)$
  Compute ensemble predictions $\mathcal{P}_e$ by comparing probabilities $\mathbb{P}_e$ with discrimination thresholds δ_e
  and append it to $\mathcal{P}_E$
end

3.4 Model Performance Metrics

In order to compare and rate the performances of various models, we need a method to evaluate them. These are usually done using metrics. Metrics vary for

classification and regression problems. In a binary classification problem, there are standard metrics and guidelines on reading those metrics that we will review in this section.

3.4.1 Confusion Matrix

A confusion matrix is a table used to describe the performance of a classification model on a set of test data for which we have ground truth. It acts as a summary of how the model did on the test data. A simple confusion matrix and its components for a binary classification task are shown in figure 3-2.

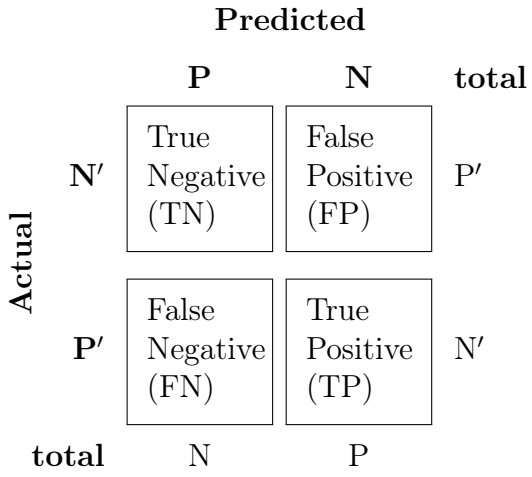


Figure 3-2. A simple confusion matrix for binary classification task

Here,

- P' and N' are the ground truth value and P and N are the outcome predicted by the model.
- True positives (TP) are outcomes where the model *correctly* predicts the *positive* class.
- True negatives (TN) are outcomes where the model *correctly* predicts the *negative* class.
- False positives (FP) are outcomes where the model *incorrectly* predicts the *positive* class.

- False negatives (FN) are outcomes where the model *incorrectly* predicts the *negative* class.

Ideally, we want to maximize the true positives and true negatives and minimize false positives and false negatives. The type of problem dictates which outcome is more important.

3.4.2 Accuracy

Accuracy is a standard metric for used classification and is the fraction of the predictions that the model correctly predicted. Equation 3.1 shows the formula for calculating accuracy in a binary classification problem.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

While accuracy is a standard metric, it does not lend itself well for measuring model performance in unbalanced datasets commonly encountered in the medical domain [78], [79]. For example, if one class is 90% more prevalent than another class, a model can just predict the frequent class to get an accuracy of 90%. For this reason, we do not use accuracy as a primary metric for our tasks.

3.4.3 Sensitivity

Sensitivity (also called recall) measures the proportion of actual positives that are correctly identified. Equation 3.2 shows the formula for calculating sensitivity in a binary classification problem.

$$Sensitivity = \frac{TP}{TP + FN} \tag{3.2}$$

In general, if correctly identifying positives is important to us, we would use a model that had high sensitivity.

3.4.4 Specificity

Specificity measures the proportion of actual negatives that are correctly identified. Specificity can be thought of as sensitivity for the negative class.

Equation 3.2 shows the formula for calculating specificity in a binary classification problem.

$$Specificity = \frac{TN}{TN + FP} \tag{3.3}$$

In general, if correctly identifying positives is important to us, we would use a model that had high specificity.

3.4.5 Positive Predictive Value

Positive Predictive Value (PPV) (also called precision) measures the proportion of correctly identified positives. Equation 3.4 shows the formula for calculating PPV in a binary classification problem.

$$PPV = \frac{TP}{TP + FP} \tag{3.4}$$

We can trust the predictions of a model with higher PPV than the predictions of a model with lower PPV, as the model with higher PPV is more likely to be correct in its predictions of the positive class.

3.4.6 Area Under the Receiver Operating Characteristics

Binary classification models, in general, output a probability that a particular data point belongs to a particular class. By comparing that probability against a *discrimination threshold*, we can predict which class that data point belongs to.

A natural choice for a discrimination threshold is 0.5, as it falls right in the middle of either extreme of the probability range. However, this discrimination threshold is dependent on the task. Furthermore, the value of this threshold determines all of our other metrics. The receiver operating characteristic (ROC) curve illustrates the diagnostic ability of a binary classifier as its discrimination threshold is varied.

The ROC curve contains the false positive rate (FPR) on its x-axis and sensitivity on its y-axis. Here, $FPR = 1 - Specificity$. The area under the ROC (AUC) curve is a value between 0.5 and 1 which makes it easy to compare multiple ROC curves. A higher AUC score implies better model performance.

3.4.7 Regression

All of the metrics seen in the above sections are for classification problems and are not applicable to regression problems, where a target variable is a real number. In this case, we want to use an error function that measures how different the predicted value is from the target value. Unlike some of the previous metrics, when we use an error function, we desire lower values, as a lower value of error corresponds to better model prediction and model performance.

Root mean square error (RMSE) is a commonly used metric that takes the square root of the average error squared between the predicted value and the actual value. This is given by equation 3.5

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - a_i)^2} \tag{3.5}$$

In some cases for numerical stability, the logarithm of the predicted and actual values are given instead of the original values. This is known as root mean square log error (RMSLE) given by equation 3.6.

$$RMSLE = \sqrt{\frac{1}{n}\sum(\log(p_i + 1) - \log(a_i + 1))^2} \tag{3.6}$$

3.5 Discrimination Threshold Selection

As mentioned earlier, the discrimination threshold is the single most hyperparameter for a binary classification model, as its value determines the values of all other metrics and hence the performance of our model. Unfortunately, the selection of the discrimination threshold is highly dependent on the task. While the ROC curve gives us a graphical view of our model's performance as the discrimination threshold is varied, it does not provide an optimal discrimination threshold. Koyejo et al. [80] propose two algorithms to find the "optimal classifiers as the sign of the thresholded conditional probability of the positive class, with a performance metric-dependent threshold". However, in their work, each metric is dependent on a separate threshold and there is not a global threshold, which would be more ideal.

For binary classification tasks predominantly seen in the medical domain, sensitivity, specificity, and PPV are the primary metrics. Thus, we want a discrimination threshold that maximizes these values. A common approach is to fix the desired value for sensitivity and vary the threshold to maximize the specificity for that value of sensitivity.

If sensitivity and specificity are both equally important, a more sophisticated approach is to use the *Youden index* [81]–[84]. The Youden index [J] is given by equation

$$J = Max_c(Sensitivity_c + Specificity_c - 1) \tag{3.7}$$

where, c is the discrimination threshold. The maximum value of the Youden index is 1, which is a perfect test, and the minimum value is 0 when the test has no predictive value. The minimum occurs when $Sensitivity = 1 - Specificity$, i.e., represented by the equal line in the ROC curve. The optimal outcome of a classifier is when both sensitivity and specificity are 1. "The point representing this combination will be in the upper left corner of the ROC curve. The closer a ROC curve is to this ideal situation, the better the classifier performs, given that both sensitivity and specificity are of equal importance" [83].

While, the *Youden* index balances sensitivity and specificity, the F1 score balances sensitivity and PPV. In our work, we use the *Youden* index as a guide to determine the optimal discrimination threshold while striving to achieve the maximum sensitivity. Given that we are interested in sensitivity, specificity, and PPV, it would be interesting to see how they vary as a function of the discrimination threshold. As such, we also see how these scores vary with the discrimination threshold as an additional guideline for selecting an optimal value.

CHAPTER 4
CASE STUDY: MERCARI PRICE SUGGESTION

To showcase the potential of leveraging both structured and unstructured data, we start with a case study involving a task from the consumer market domain. It is important to note that while our inspiration for this work comes from the healthcare domain, we want to demonstrate the generalizability of our framework by applying it to a non-healthcare domain.

We use a Kaggle [1] dataset that was published as part of a competition and contains both structured and unstructured data. The dataset belongs to a company *Mercari* and is available as part of the *Mercari Price Suggestion Challenge*. Mercari is Japan's biggest community-powered shopping app. Sellers post items that they want to sell along with an asking price. Mercari offers pricing suggestions to them. The objective of this task is to build a model that automatically suggests the right item prices based on input data. Throughout this chapter, we refer to this dataset as the *Mercari dataset*.

4.1 Mercari Dataset

The dataset provided by Mercari for this task is available for download from the Kaggle website. The dataset is provided in two tab-separated files. One is the training set and consists of all input data along with the price of the item which is the dependent variable. Another file is the testing set that contains only the input data and does not include the price of the items.

This dataset contains both structured and unstructured data. Structured data contains categorical variables. Categorical variables represent a category usually expressed as strings in the data which has to be preprocessed so that it can be used by the model. Table 4-1 provides an overview of the dataset.

[1] https://www.kaggle.com/

Table 4-1. Overview of the Mercari Dataset

Name	Cardinality	Missing	Details (from Kaggle)
train_id	N/A	0	The ID of the listing
name	N/A	0	The title of the listing.
item_condition_id	5	0	The condition of the items provided by the seller
category_name	1287	6327	Category of the listing
brand_name	4809	632682	N/A
shipping	2	0	1 if shipping fee is paid by seller and 0 by buyer
item_description	N/A	4	The full description of the item.
price	N/A	0	The price that the item was sold for. This is the target variable that you will predict. The unit is USD.

Since this is a regression problem the root mean squared logarithmic error (RMSLE) given by equation 3.6 is the primary metric to evaluate the model. We repeat it here for convenience,

$$\epsilon = \sqrt{\frac{1}{n}\sum(\log(p_i + 1) - \log(a_i + 1))^2} \tag{4.1}$$

where,

- ϵ is the RMSLE value (score)
- n is the total number of observations in the dataset
- p_i is the price predicted by the model for item i
- a_i is the actual sale price of item i
- $\log(x)$ is the natural logarithm of x

In order to evaluate our model on the test set, we upload our code (also known as a kernel) to the Kaggle server and commit it. This runs our code directly on the server and outputs a prediction file. This prediction file is then scored according to the RMSLE and a final score is provided.

While we are interested in the performance of our model in the held-out test set, we also want to evaluate statistically the performance of the model over multiple iterations of the training set. Thus, in addition to providing the RMSLE of the model on the held-out test set, we also provide the average RMSLE of the model over 100 iterations of the training and validation set split with an 85%-15% split.

4.2 Data Exploration and Preprocessing

In order to understand the dataset, we perform explanatory data analysis on each variable. Note that for those analyses which depend on the price of an item, we can only use the training set as the testing set does not contain the price variable. For those analyses which do not depend on the price of the item, all details reported were based on the entire available dataset including the training and testing datasets.

We also need to preprocess the data in order for it to be used by the model. We report the preprocessing of the variables as we explore it.

4.2.1 Price

This is the target variable that must be predicted by our model. The price is listed in USD. Mercari does not allow postings less than $3 and there only 3 values that are over $2000 which are likely shipping costs. Thus we only include entries that have prices within the $3 and $2000 range. Furthermore, since the competition is judged based on the RMSLE, we apply a *log one plus* ($log1p$) to the target variable. The one is added for numerical stability.

It should be noted that by doing this, our model is trained to predict a value in the $log1p$ range, thus to get the actual price of the item, we need to apply na *exponential minus one* ($expm1$). Figure 4-1 shows the distribution of the prices across all the items from the training set. We can see that most of prices fall less than $100. As expected the $log1p$ distribution is similar to the original distribution.

4.2.2 Shipping

Shipping can either be paid by the seller or the buyer. This value is represented by a 0 or a 1. In case the shipping is unspecified in the dataset, we assume that it was paid by the buyer. We found that the shipping variable was almost evenly split across all the datasets. Shipping was paid by the seller for about 55.2% items

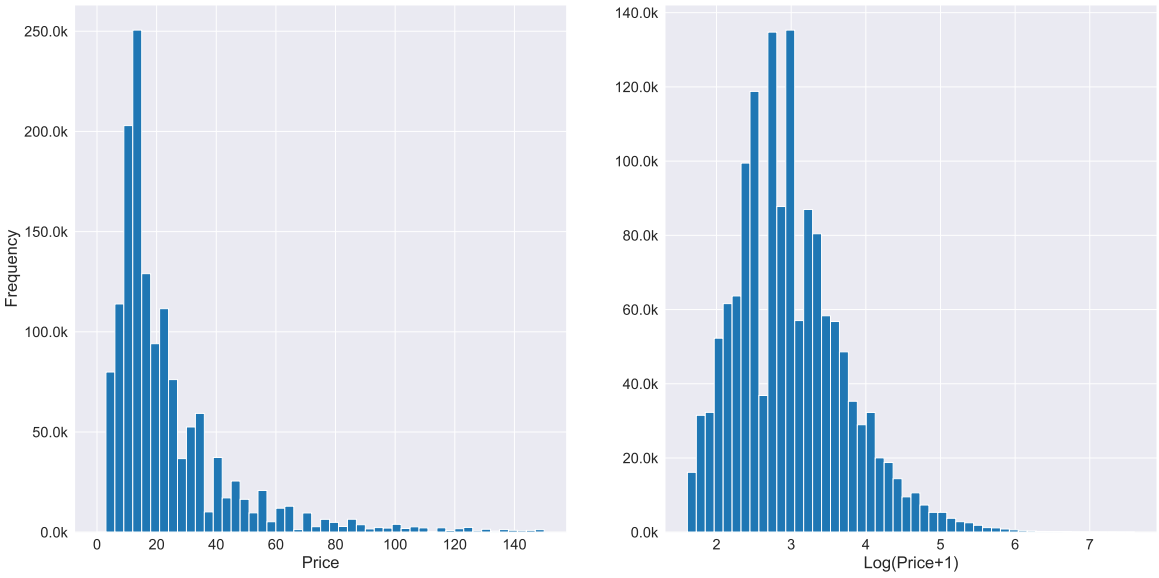


Figure 4-1. Price distribution of the items: (Left) Price value in USD. (Right) Log 1 plus of price value

and was paid by the buyer for about 44.8% items. In general, the average price paid by the users who have to pay for shipping fees is lower than those that do not require an additional shipping cost. This can be seen in figure 4-2 and matches our perception that the sellers need a lower price to compensate for the additional shipping cost.

4.2.3 Item Category

Item categories are represented as strings and include a main category and two sub-categories separated by a backward slash. For example the string *Beauty/Makeup/Face or Lips* has *Beauty* as the main category and *Makeup* and *Face or Lips* as subcategories. During our preprocessing, we split the item category into *main_cat*, *sub_cat1*, and *sub_cat2*. If any item category is missing, we just replace them with a string "missing" to indicate it.

There are 1287 unique item categories which include 11 main categories, 114 first subcategory and, 905 second subcategory. Figures 4-3, 4-4, and 4-5 show the distribution of the main category and top 15 of the first and second subcategories respectively across all items. We can see that most of the items belong to the main

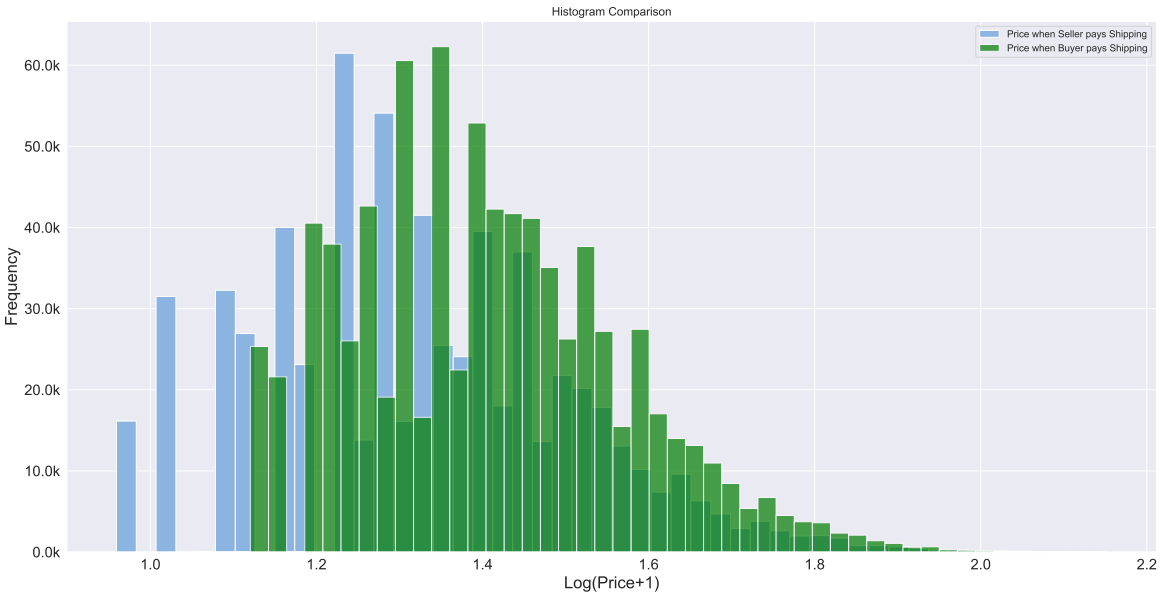


Figure 4-2. Price distribution of the items grouped by shipping

category *Women* thus being skewed to the left, while both the subcategories are more diversely distributed across a range of categories.

In order to convert categorical variables into numerical features, we used *mean target encoding*. Mean target encoding uses the target variable as the bases to generate the new encoded feature. In other words, mean target encoding represents the probability of the target variable, conditional on each value of the feature. This is done by assigning a value for a categorical variable that is a function of the mean of the target variable for that category. For more details about how mean target encoding is applied to this dataset, please refer to listing A.1 and A.2 in appendix A. All the categories are converted to numerical values using this method.

4.2.4 Brand Name

Brand names are given as strings and there are 6312 unique brand names across all items. The training dataset has only 4791 unique brands indicating that there are unseen brands in the testing sets. For items with no brand names, we replace it with the string “missing“ to indicate it. Figure 4-6 shows the distribution of the top 15 brands across all items. It is clear that brands that are not available or missing account for a large percentage of brand category. The brand name is a categorical variable and is encoded into a numerical value using mean target encoding. It is

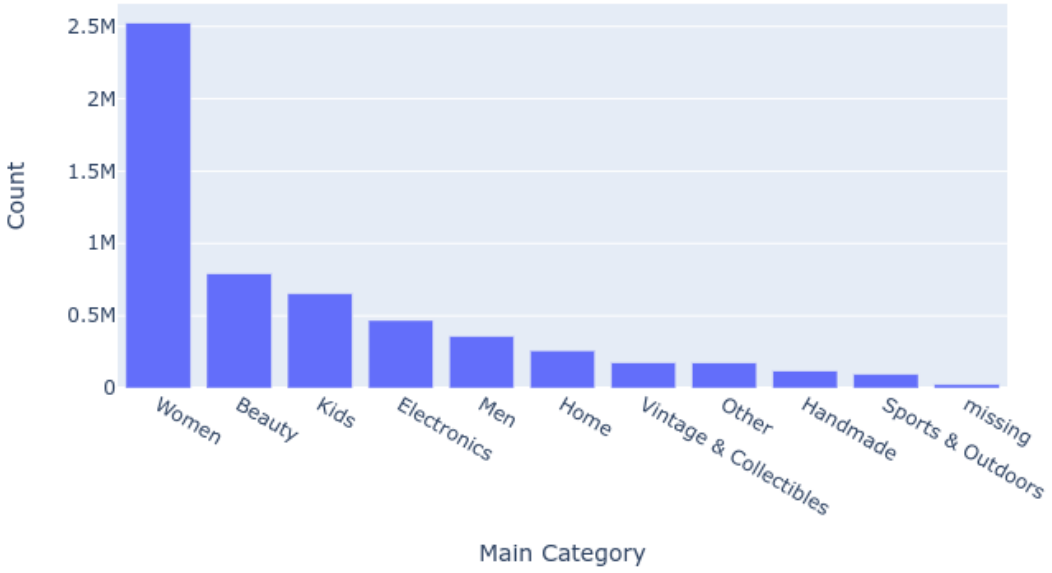


Figure 4-3. Distribution of main category across all items

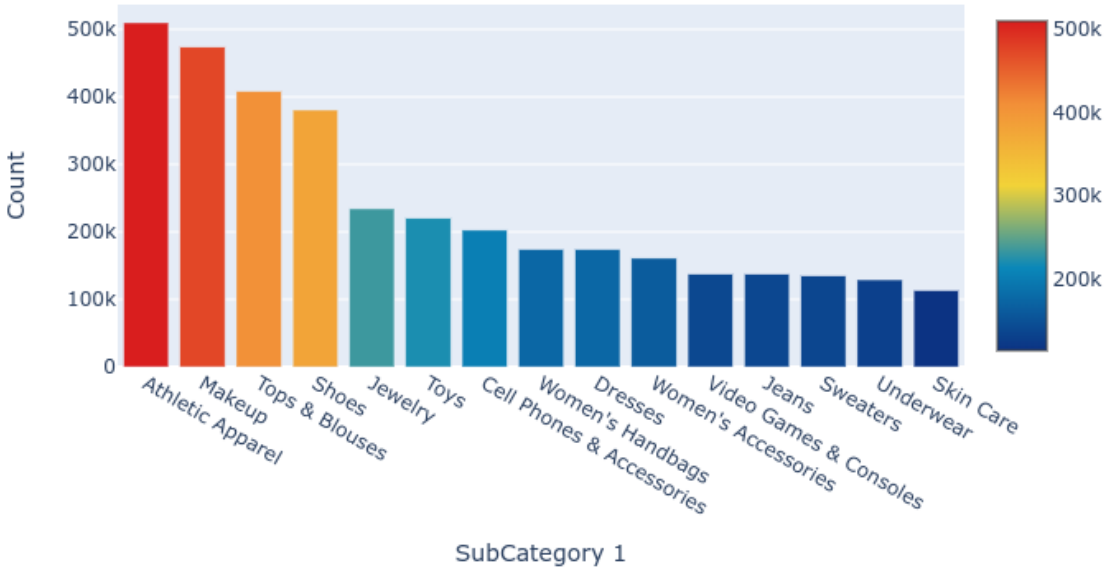


Figure 4-4. Distribution of the top 15 first subcategories across all items

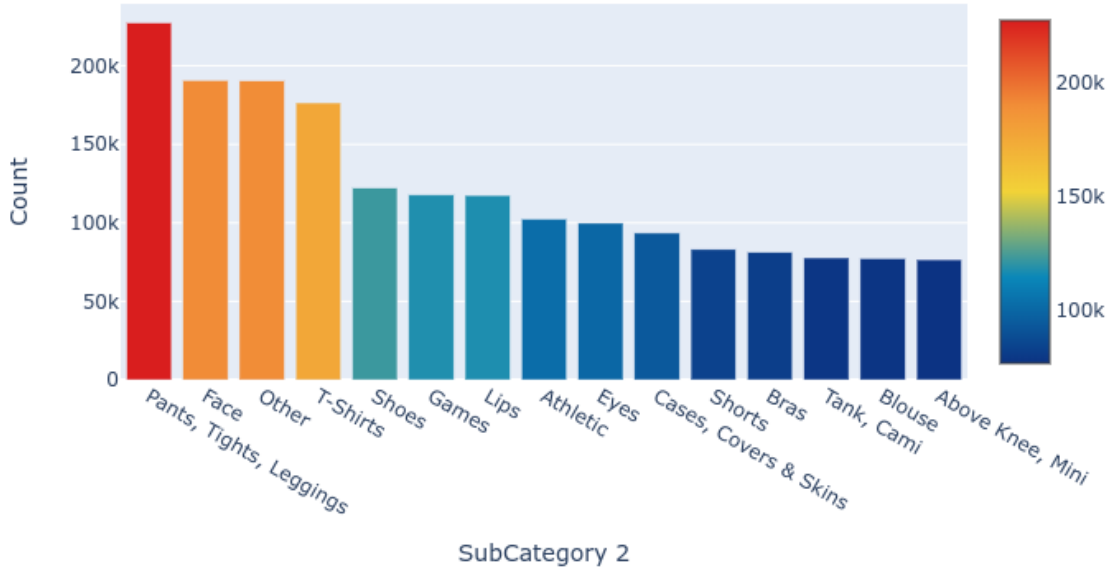


Figure 4-5. Distribution of the top 15 second subcategories across all items

clear that brands that are not available or missing account for a large percentage of brand category.

4.2.5 Item Name and Description

Item names and descriptions are user input text and are treated as unstructured data. For simplicity, item names and descriptions are concatenated together. Missing names or descriptions are replaced by empty strings. Figure 4-7 shows the distribution of the item description length across all items. As expected the majority of item descriptions are short, running less than 30 characters long.

In order to vectorize the item description, we applied a bi-gram TF-IDF vectorizer [69] provided by the Scikit-Learn library [71]. We used a maximum vocabulary size of 60000. It is important to note that, we did not perform commonly used preprocessing such as removing punctuations or replacing words with their lemmas. Our experiments showed that using raw data gave better performance.

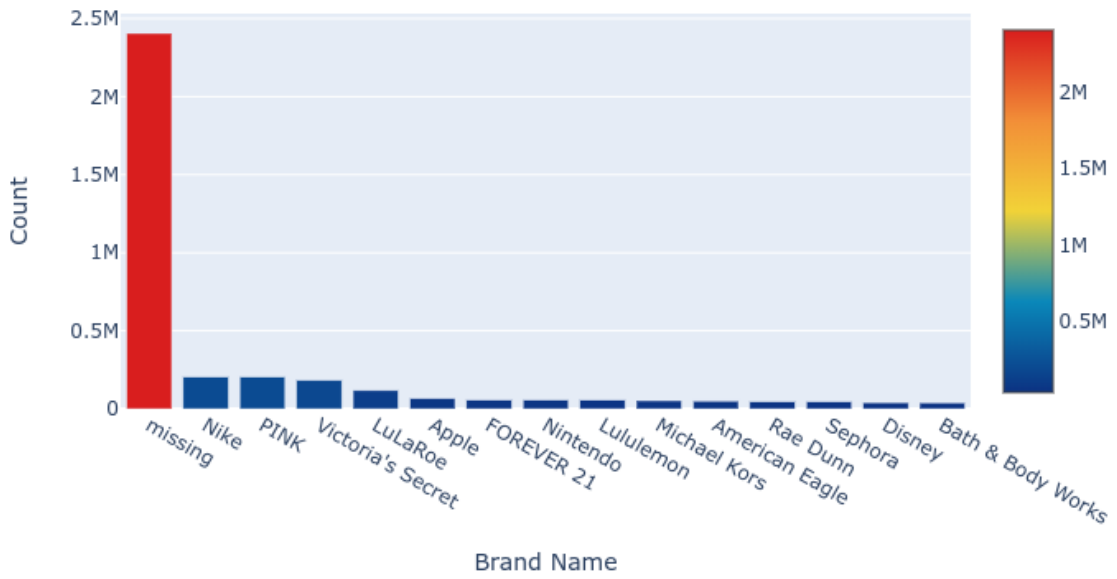


Figure 4-6. Distribution of the top 15 brands across all items

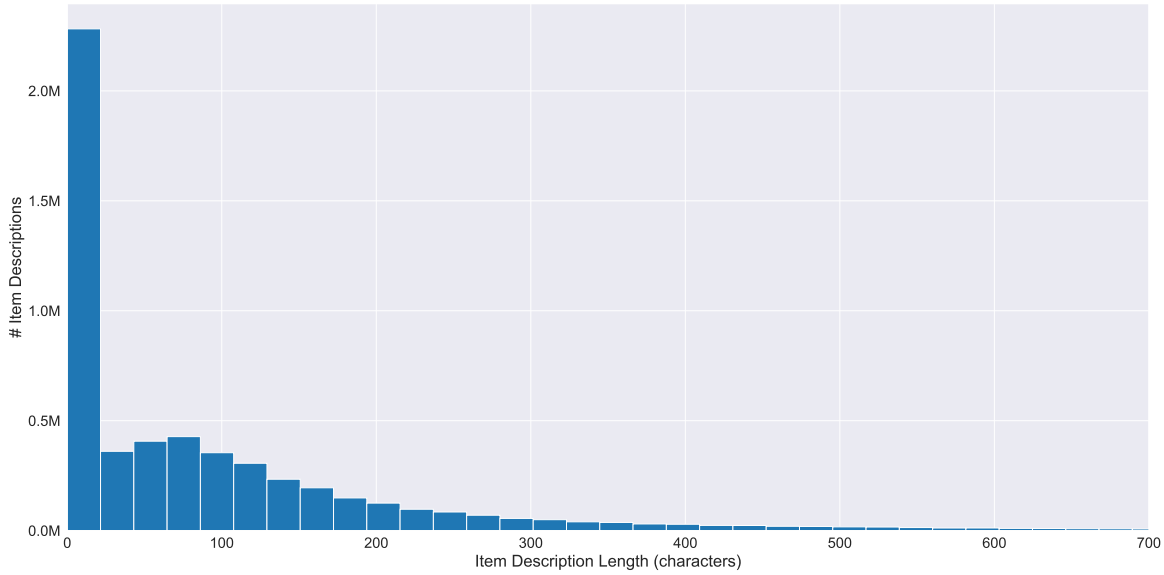


Figure 4-7. Distribution of the item description length across all items

4.3 Results

We built a gradient boosting model [74] using the LightGBM [75] software library. We used the training dataset to split it into a training and validation set with an 85%-15% split. We used this training subset for training our model and the validation set to evaluate the model's performance and to perform cross-validation to tune the hyperparameters of the model. Please refer to listing appendix section A.3 for model details.

Once we chose the hyperparameters, we ran 10 iterations where we used a distinct seed to split the data into a training set and test set, built models on the training set, and obtained the RMSLE on the test set. We did this for 3 subsets of the data, one using only structured data, one using only unstructured data, and one using both structured and unstructured data. The average RMSLE out of the 10 iterations for each type of data is shown in figure 4-8 as a point plot.

A word cloud of all the tokens that the model thinks are important and the feature importance of the structured data relative to the most important feature is shown in figures 4-9 and 4-10 respectively.

4.4 Discussion

We notice that using only the structured data gives us an average RMSLE of 0.541 over 10 iterations. However, using only the unstructured data decreases the RMSLE by over 13% to 0.469. This shows that the information held in unstructured free text is valuable and adding it to the structured data could enhance model performance. This is demonstrated by the fact that by using both types of data we are able to get an RMSLE of 0.439 which is over 20% less than using only structured data and 8.5% less than using only unstructured data. These results indicate the potential of integrating multimodal data in increasing model performance.

Figure 4-10 indicates that *brand_name* is the most important feature to the model. This is reinforced by the word cloud where phrases such as *lularoe* and *lululemon* are rated as important which are top-rated brands of clothes. Despite the majority of the items having the brand name as unavailable or missing, we can see brand names is a very important factor to determine the price of the time, which is both supported by the relative feature importance given to them by the model in both the structured data and the item description. It is important to note that

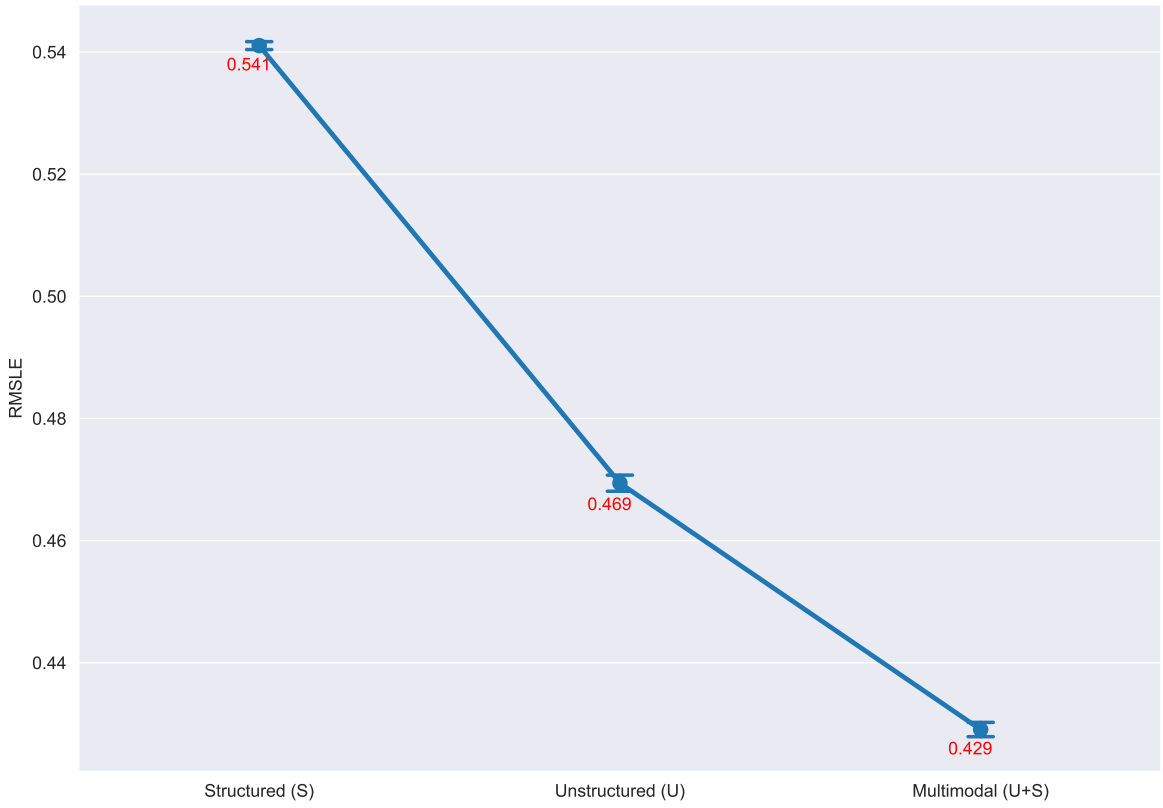


Figure 4-8. Average RMSLE over 10 iterations for each type of data

analyzing word clouds and feature importances are an informal and non-scientific way to understand what the model thinks is important.

To conclude, in this case study we were able to use our pipeline in a consumer market domain to explore the potential of integrating multimodal data and how it can affect machine learning model performance. Furthermore, the generalizability of the pipeline will be emphasized when we apply it to a medical domain in the upcoming chapters.

Figure 4-9. Word Cloud of the top 500 tokens produced by the model

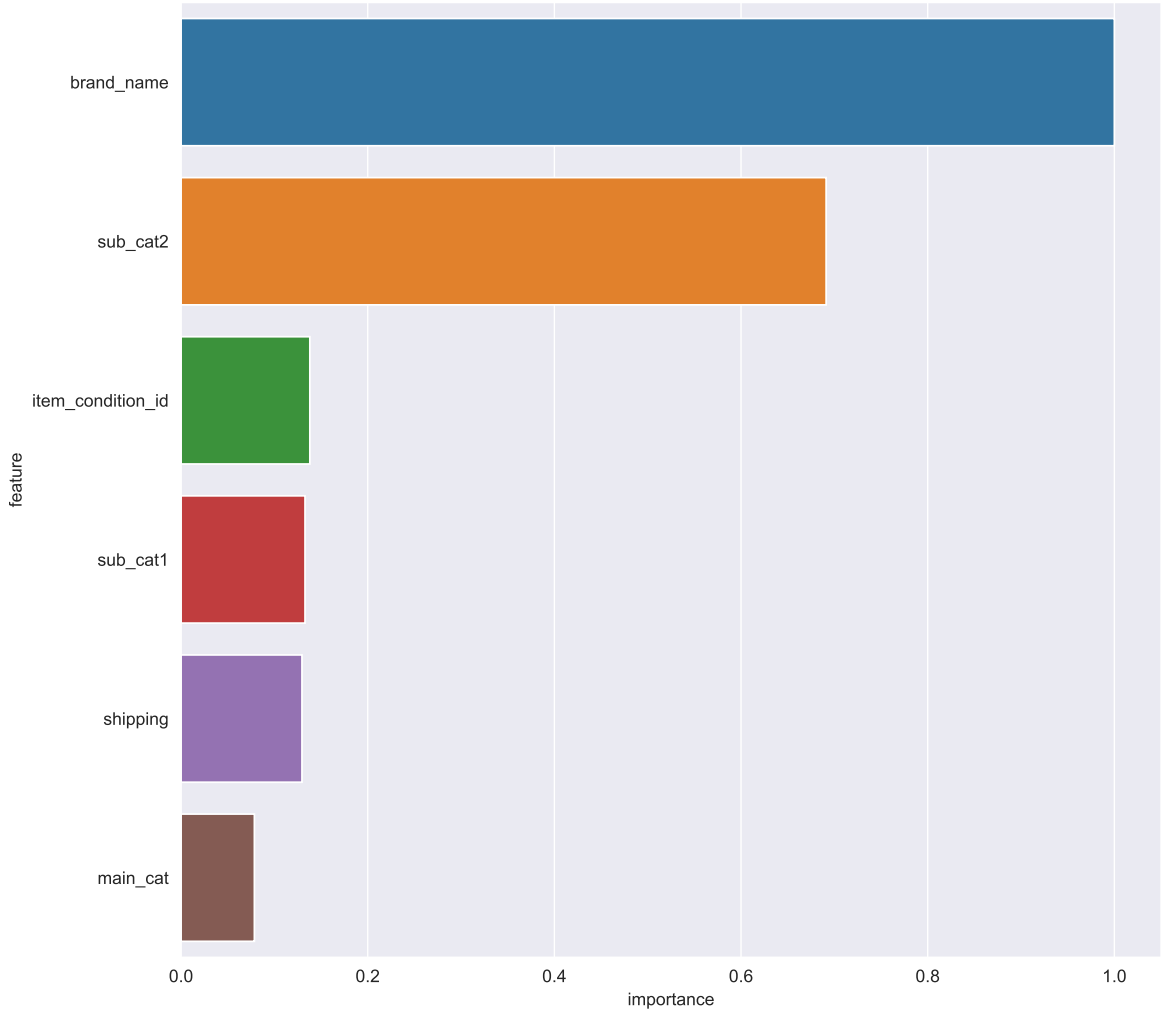


Figure 4-10. Feature importance relative to the most important feature

CHAPTER 5

CASE STUDY: PREDICTING IMMINENT ICU ADMISSION USING MIMIC DATASET

In this chapter, we switch gears from the consumer market domain to the healthcare domain in applying our pipeline for building ML models integrating multimodal data. Compared to the previous task, healthcare data is more complicated as there is a distinct time component. Specifically, a patient's status is heavily dependent on their medical history, which has to be accounted for while building ML models for a particular task.

5.1 Imminent ICU Admission Prediction

Critically ill patients requiring acute care management are admitted to the Intensive Care Unit (ICU). A myriad of parallel processes occurs well before the patient is admitted to the ICU. Each of these processes generates significant volumes of data, namely a variety of records usually consisting of vital signs recordings, any laboratory values, and the clinical staff assessments, recorded as unstructured text which, in turn, informs further clinical decision making. The evidence-informed decision making involved in the admission of a critically ill patient to the ICU can be dynamic and complex.

Recent data, however, has indicated that most patients experience delays in ICU admission [85], [86]. Indeed, for hospitalized patients who rapidly deteriorate, slow transfers to the ICU has been associated with increased risk of death, cost, and such patients were less likely to receive physician bed-side evaluation within 3-hours of documented deterioration [87]. Delayed patients were also found to have a greater requirement for advanced respiratory support, often experiencing longer ventilator days [88]. Each hour of delay in the ICU admission was associated with a 1.5% increase in the risk of ICU death [89]. Therefore, earlier identification of patients who meet criteria for admission to the ICU may reduce the likelihood of death, costs, and improve long term patient outcomes. The use of unstructured

data may identify novel features of deterioration that complement clinical intuition at the bedside.

Clinical notes constitute the majority of unstructured data. These are written by healthcare providers during patient visits. The clinical language in these notes reflects the medical status of the patient. Natural language processing (NLP) enable terms in notes to be included in models [90]. Traditional indicators of ICU admission related tasks have largely been physiological indicators [91] or clinical observations [92]. Common laboratory test results have been used to predict imminent emergency team calls and ICU admission, ICU readmission [93], or death [94]. Severity scores such as quick Sepsis-related Organ Failure Assessment (qSOFA) and Acute Physiology and Chronic Health Evaluation (APACHE) II have also been used to predict ICU admission in patients with clinically diagnosed infection [85], [95], [96]. While most of the data used in these studies are part of the structured data stream [86], [97], there is less known about the use of clinical notes for predicting imminent ICU admissions. Clinical notes, however, have been effectively used in compliment tasks to imminent ICU admission prediction, although they have been used for tasks that occur after ICU admissions such as [40], [98], [99], and prolonged length of stay prediction [100].

Following the same structure as the previous task, in this work, we use both structured data in the form of vital signs measurements taken for the patients, as well as unstructured data in the form of clinical notes that were recorded by healthcare professionals to predict whether the patient is likely to be admitted to the ICU in the next 24-48 hours. Specifically, we define imminent ICU admission as a condition that may require an ICU admission in the next 24-48 hours.

5.2 MIMIC Dataset

Medical Information Mart for Intensive Care version 3[1](MIMIC-III) [48] is a large, single-center database comprising information relating to patients admitted to critical care units (ICU) at Beth Israel Deaconess Medical Center in Boston, Massachusetts. It contains data associated with 58,976 distinct hospital encounters for 46,520 patients admitted to critical care units between 2001 and 2012. MIMIC III has been used as a research dataset for a wide variety of clinical tasks including

[1]https://mimic.physionet.org/

predicting length of stay [101], [102], mortality prediction [47], [49], [103], [104] and clinical time series analysis [105]–[107].

The dataset is stored in a relational database consisting of 26 tables with information regarding admissions, discharges, patients, procedures, prescriptions, and diagnosis (table 4 of [48]). Information about associated code can be found in [108] and is available in Github[2]. The dataset contains both structured and unstructured data. Structured data includes information such as patient vitals, chemical levels, and lab test results. Clinical notes grouped by categories written by nurses, physicians, and other healthcare personal account for unstructured data. We will refer to this dataset as the *MIMIC dataset*.

In this chapter, using the MIMIC dataset, we determine whether the health of a patient admitted to a hospital has deteriorated to an extent that requires immediate ICU admission. In particular, we perform 3 different experiments. First, we use only structured data (vitals) to build ML models, then we use unstructured data (clinical notes), and finally, we integrate both these types of data and feed in the multimodal data to build a classifier to predict the same task. By comparing the performances of the models across a wide range of metrics over different iterations, we are able to demonstrate the strength and potential of using multimodal data to build predictive models.

5.3 Data Setup

We start by defining certain terminologies. An *encounter* for a patient is defined as any situation where the patient visits a hospital. The patient may stay at the hospital for many days and have data gathered about their stays such as clinical notes, vital signs, and lab results, etc at multiple times. All these belong to the same encounter. Each unique encounter contains details about a single patient. An encounter is the basic unit of the identity of the data, and all subsets (including training and testing sets) are partitioned according to encounters.

During a single encounter, the patient may have to be admitted to the ICU. This is known as an ICU admission. There may be multiple ICU admissions during a single encounter, however, for this work, we only consider the *first* ICU admission of an encounter.

[2] https://github.com/MIT-LCP/mimic-code

5.3.1 Data Labeling

In order to obtain a labeled dataset for our task, we partition the time between hospital encounter and first ICU admission and label each partition. In particular, we consider the time of ICU admission as time $t = 0$ days. All data recorded 24 hours prior to the ICU admission is discarded. This is because our objective is to predict whether there is imminent ICU admission in the next 24-48 hours. The data recorded between time $-3 \leq t \leq -1$ days belong to the positive class pertaining to an imminent ICU admission. Next, the data recorded between time $-5 \leq t \leq -3$ days are discarded due to potential data leakage between the two classes. Finally, all the data recorded before $-5 \leq t \leq -15$ days belong to the negative class pertaining to delayed ICU admission. Figure 5-1, shows an overview of how the data is labeled.

5.3.2 Data Filtering

In order to form a cohort for our tasks, we applied exclusions on the patients, encounters, and the notes.

Figure 5-2 shows a flow chart of the filtering process. We only include patients who are older than 15 years and exclude those encounters that occur within 30 days for the same patient. This is because, if a patient gets admitted to the hospital within 30 days, the likelihood of the second admission being due to the same problem as the first admission is higher. Since we want our models to generalize over a wide array of medical problems, we have decided to exclude hospital encounters that occurred within 30 days of the previous encounter, since it is likely that the new encounter might be for the same medical problem.

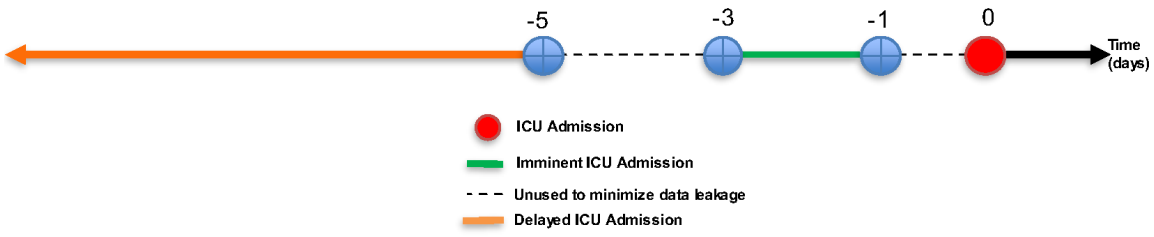


Figure 5-1. Timeline showing data labeling recorded between hospital encounter and first ICU admission

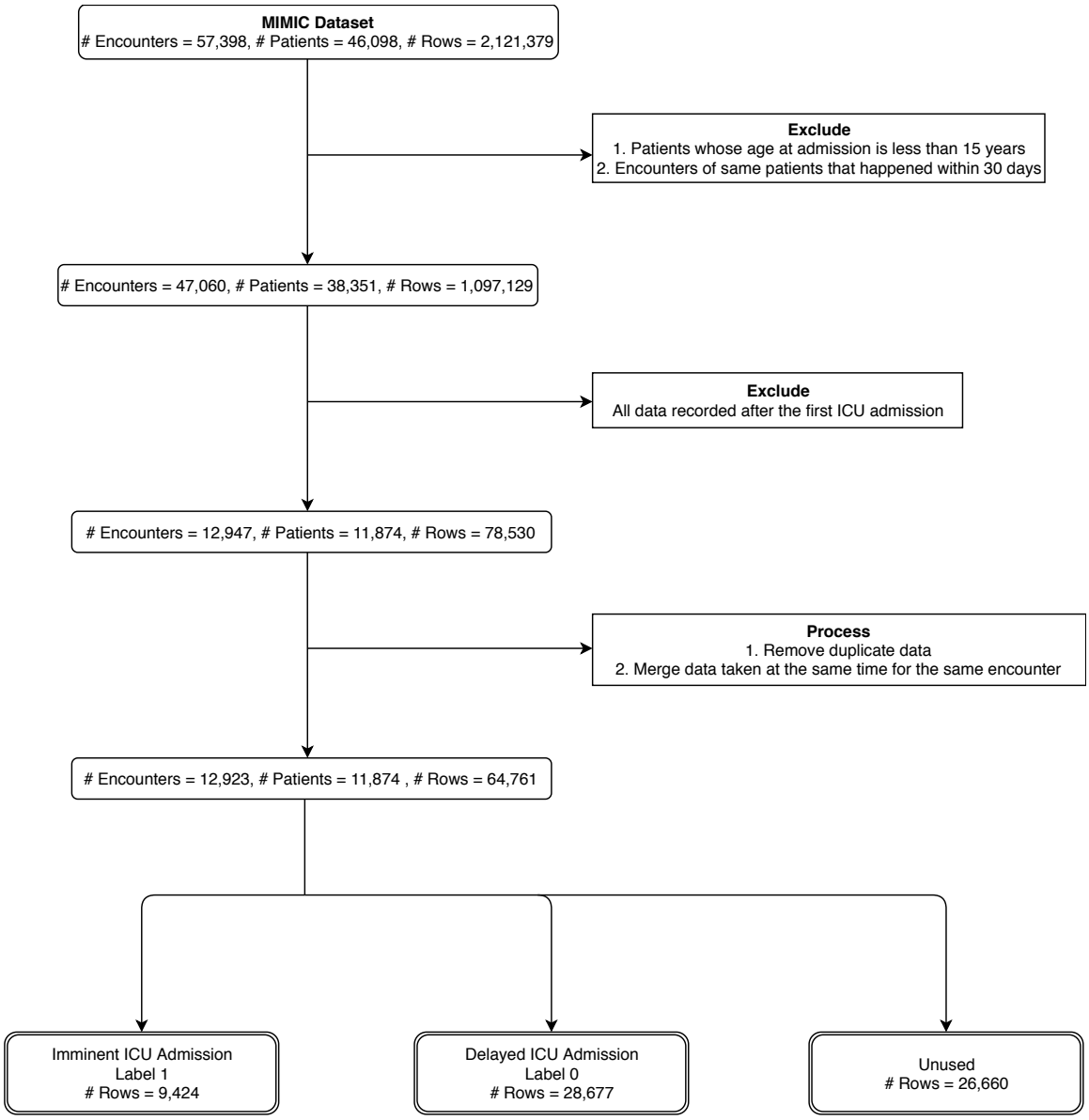


Figure 5-2. Flow chart of how the data is filtered

As mentioned earlier, we are only interested in predicting whether there is an imminent ICU admission for the first ICU admission. Thus all the data recorded after the first ICU admission is irrelevant, which can be excluded. Finally, we found out there were some duplicate data recorded for patients and some data recorded at the same time for the same encounter. This is mainly true for clinical notes. The duplicate data were removed and the data taken at the same time were either merged or discarded depending on the type of data. In order to compare performances between different subsets of the entire datasets corresponding to the subset of data use, we created 3 different data subsets for building the models. These are called *structured*, *unstructured*, and *multimodal* data respectively corresponding to the type of data utilized by the models.

5.4 Data Exploration

We do some preliminary explanatory data analysis to explore the data. One of the important characters of text, in general, is its length. Figure 5-3 shows a histogram of the length of notes in characters. We can see that the majority of the notes are less than 2000 characters in length.

Our data labeling involves partitioning the time between note record time and ICU admission time. Figure 5-4 shows the distribution of time between note record time and ICU admission time. We can see that most of the notes were recorded within 10 days prior to ICU admission.

We represent this same data in a different view in figure 5-5. This figure shows the distribution of notes as a function of time to ICU admission. We can see that the majority of the notes were recorded within 24 hours prior to the ICU admission. This is not surprising as the health of patients deteriorates and they are being treated more aggressively, more data about their medical state is recorded. These notes are not used for modeling giving the model the ability to detect and predict the language showing health deterioration.

We also note that a significant amount of data was recorded prior to over 15 days before ICU admission. Please note that all those notes were not charted during a single day but over a period of time. We chose to clump them together at the 15-day mark for this graphic.

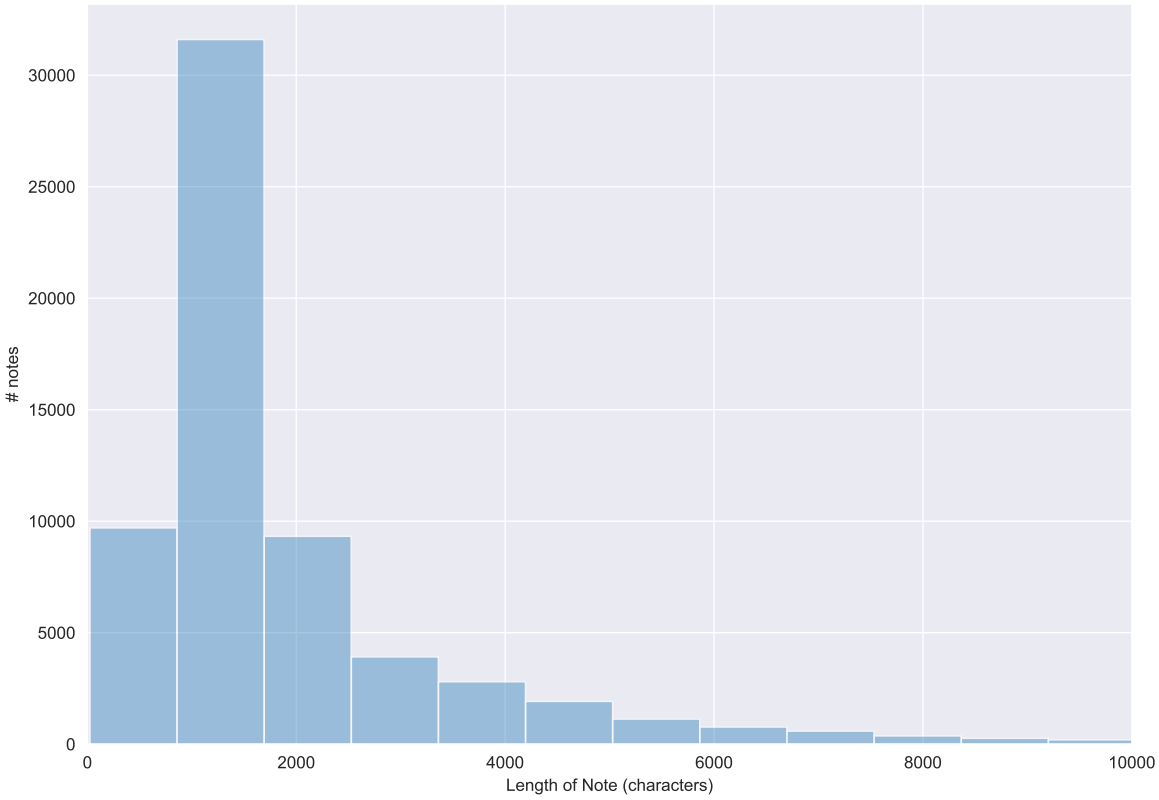


Figure 5-3. Histogram of Note Length

Finally, figure 5-6 shows the histogram of notes by the class label. Note that the *Unused* label pertains to those notes that fall outside the boundaries of the time limits we defined. These notes will not be included in building the model.

As noted earlier, we exclude data that were recorded in certain time limits. Specifically, data recorded 1 day prior to ICU admission and between 3 to 5 days prior to ICU admission are not included during modeling due to potential data leakage. Consequently, encounters that have data only in this time frame are discarded. Table 5-1 shows the characteristics of this cohort.

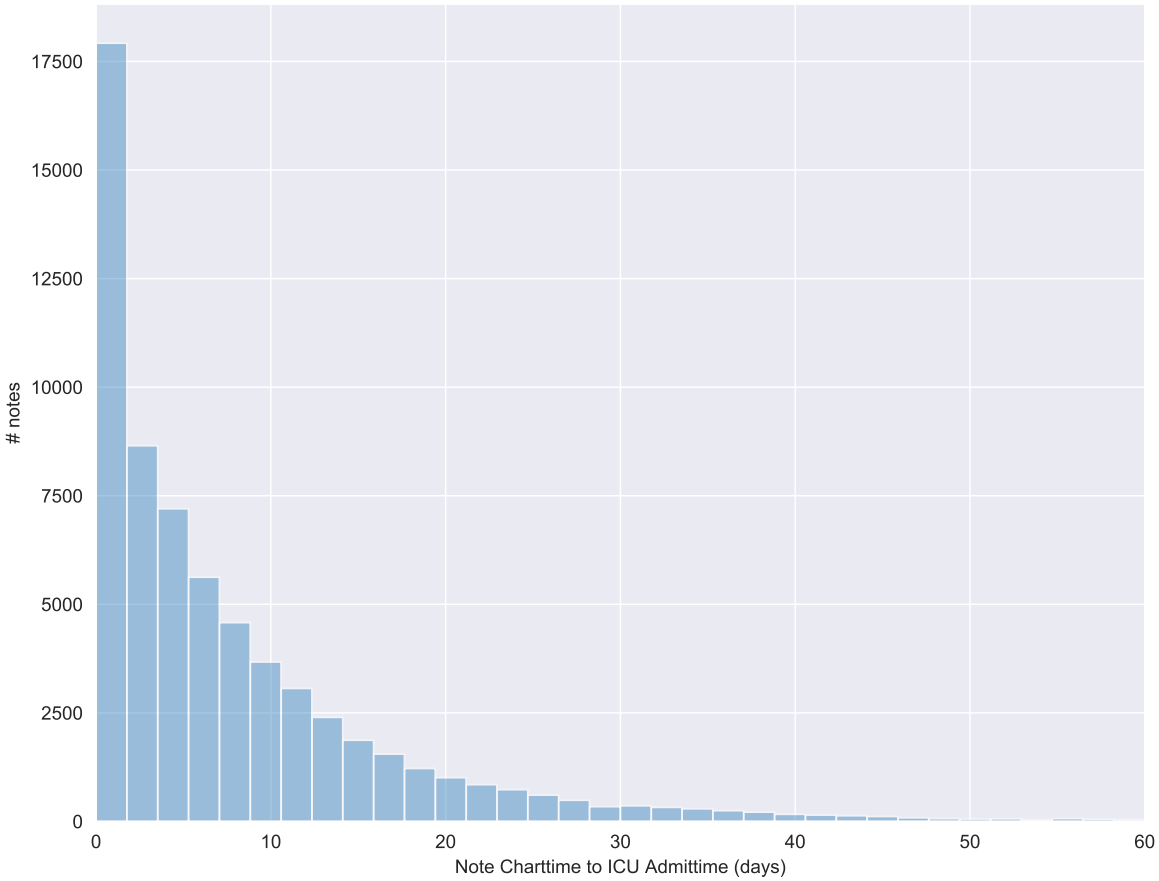


Figure 5-4. Histogram of time between note record time and ICU admission time

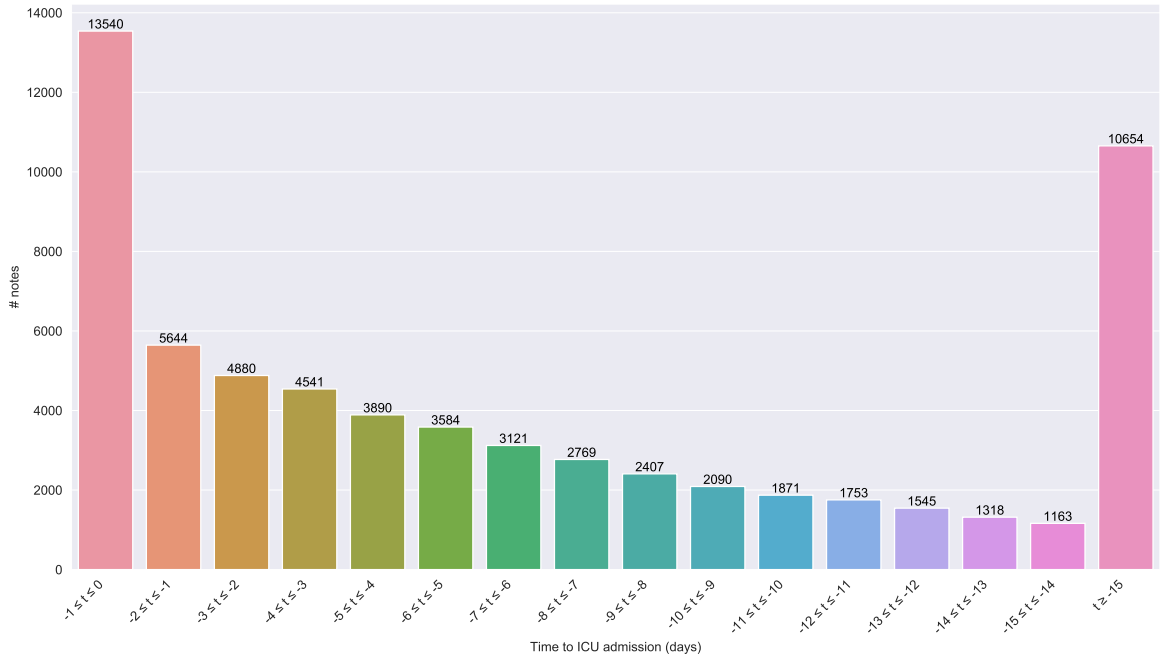


Figure 5-5. Distribution of notes with time to ICU admission

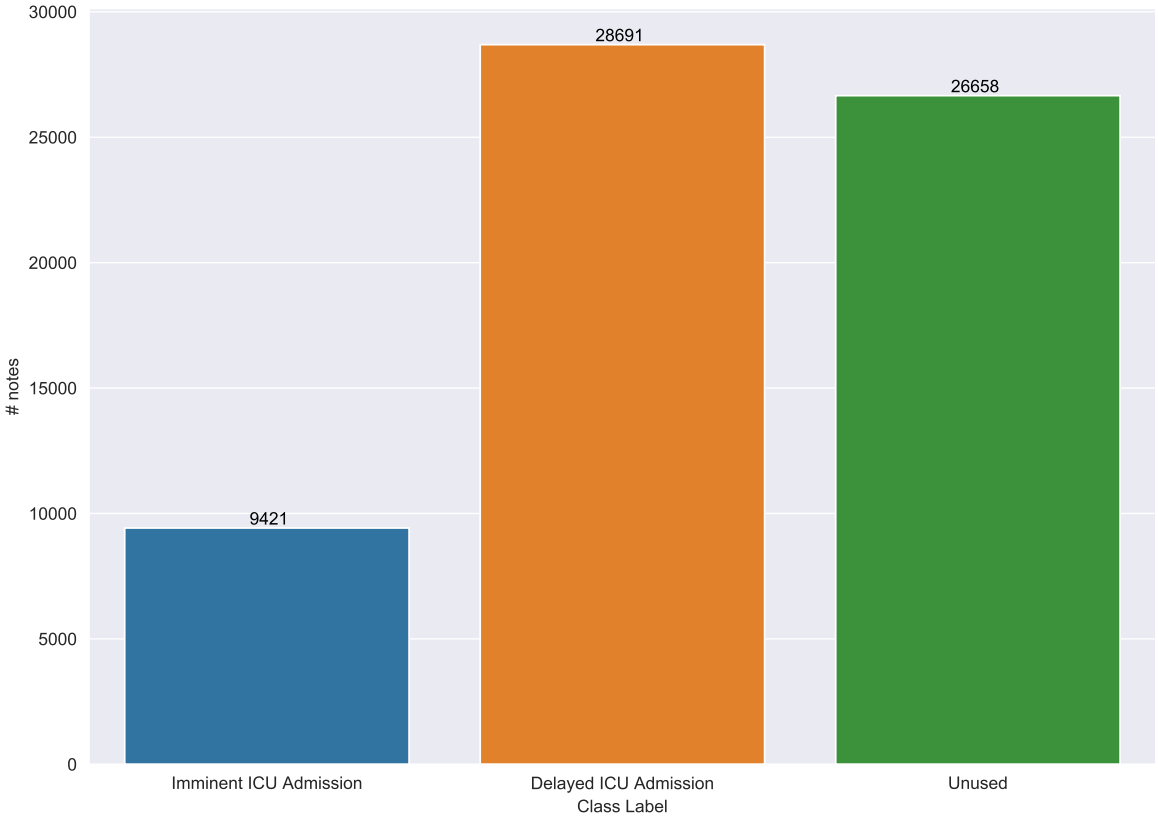


Figure 5-6. Histogram of notes by class label

5.5 Data Processing

In this section, we describe how different subsets of the data are processed. Specifically, we have 3 different processing paradigms that are enacted to this data, one each for the type of data subset.

5.5.1 Structured Data

Structured data can be in the form of category variables or real values. In this work, we include those structured variables that are represented only by real values, although the approach is easily extendable to other types of structured data. While there are many different structured variables that are routinely collected about the patient in terms of physiological indicators and clinical observations, for this work, we focus on the vital signs that were recorded for the patient as part of the structured data. Table 5-2 shows a description of the these variables.

Since healthcare data is predominantly time-based and there is a chance that not all of the variables are taken at the same time. This leads to a lot of missing data.

Table 5-1. Characteristics of the MIMIC cohort excluding unused notes

Characteristics	Value
Patients	5,289
Deaths, Number (%)	2,810 (53.1)
Male, Number (%)	3,087 (58.4)
Age, mean (SD) [IQR], y	66.3 (15.9) [56.0 – 77.9]
Time to ICU admission, mean (SD) [IQR], d	19.8 (18.3) [8.8 – 24.5]
Encounter Type, Number (%)	5,451 (100)
Elective	632 (11.6)
Emergency	4,617 (84.5)
Urgent	213 (3.9)
Ethnicity, Number (%)	
Asian	112 (2.1)
Black	367 (6.9)
Hispanic	166 (3.1)
Unknown	789 (14.9)
White	3,856 (72.9)
Average number of clinical notes per encounter	7.0
Clinical Note Length, mean [SD] [IQR]	2,665.2 [4,683.3] (996.0 - 2,310.0)
Clinical Note Category, Number (%)	38,101 (100)
Case Management	20 (0.1)
Consult	6 (0.0)
General	204 (0.5)
Nursing	2,897 (7.6)
Nursing/Other	13,025 (34.2)
Nutrition	291 (0.8)
Pharmacy	4 (0.0)
Physician	1,933 (5.1)
Radiology	18,438 (48.4)
Rehab Services	155 (0.4)
Respiratory	1,081 (2.8)
Social Work	57 (0.1)

Table 5-2. Vital Signs

Variable	Description (units)
HR	Heart rate (beats per minute)
O2Sat	Pulse Oximetry (%)
Temp	Temperature (Deg C)
SBP	Systolic BP (mm Hg)
MAP	Mean arterial pressure (mm Hg)
DBP	Diastolic BP (mm Hg)
Resp	Respiration Rate (breaths per minute)
Glucose	Serum glucose (mg/dL)

These are processed using standard data science techniques. In particular, for each encounter, missing data for each variable is forward-filled i.e., subsequent missing values are replaced with the latest recorded value. In case there are missing values for any variable before their first value, they are replaced by the population median which is standard for healthcare data.

In addition to the values of the structured variables, we also included *change statistics* of the variables over the last 24 hour period. Change statistics tracks how each variable changes and enable the model to track *patient dynamics*. For our work, we used the following 5 statistics: minimum, mean, median, standard deviation, and maximum. Thus each variable recorded at a particular time was accompanied by 5 additional values indicating how the variable had changed over the past 24 hours. In total, we have 8 vital sign data along with 5 change statistics for each variable for a total of 40 variables as part of structured data.

5.5.2 Unstructured Data

Unstructured data is composed of clinical notes written by healthcare professionals after examining a patient. It is important to note that, since these are hand-written notes their frequency is far less than those of structured data, which are often recorded by machines and are more frequently sampled.

Processing text involves two steps. First, the text is split into individual tokens in a process called *tokenization*. The tokens can consist of single words or multi-word phrases. These tokens are then converted to numbers in a process called *vectroization*. For this work, we used a simple whitespace bigram tokenizer. Here,

we just split the tokens based on whitespace separation, and each token can consist of up to two words. This method is very simple to implement. We then used the TF-IDF vectorizer to vectorize the notes with a maximum vocabulary of 60,000 tokens. We also tried more advanced techniques of tokenization and vectorization, but a simple whitespace tokenizer gave us the best performance on the end model.

5.5.3 Multimodal Data Integration

Individual subsets that we have seen are unimodal in nature i.e., they only contain the same type of data. However, when we integrate different data modalities, we get multimodal data. However, integrating these structured and unstructured data, in this case, poses a significant challenge.

Both the vital signs data and clinical notes are sampled at different frequencies and have different time intervals between each sample. Our objective is to preserve as much information from both the data sources as possible while integrating them and preparing them in a way that can be used to build ML models.

As an example, figure 5-7 shows how the different types of data are recorded in a typical encounter. The document icon represents clinical notes, while the geometric shapes represent different vital signs. As we can see, we have data that is collected at different frequencies, and within the same type of data, the time intervals are vastly different.

We are trying to solve the problem of how to integrate both these data sources into one input data that can be used to train our ML models. There are a couple of different approaches we can take to solve this problem. One approach is to

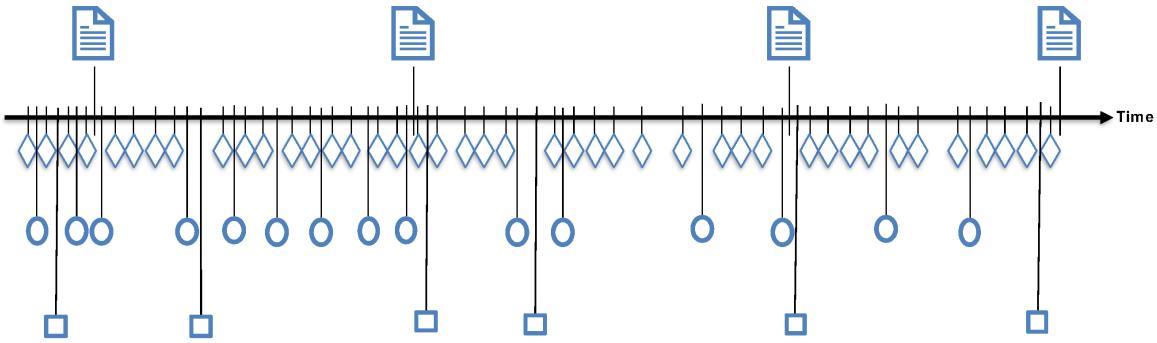


Figure 5-7. Example illustrating the complexity of integrating multimodal data

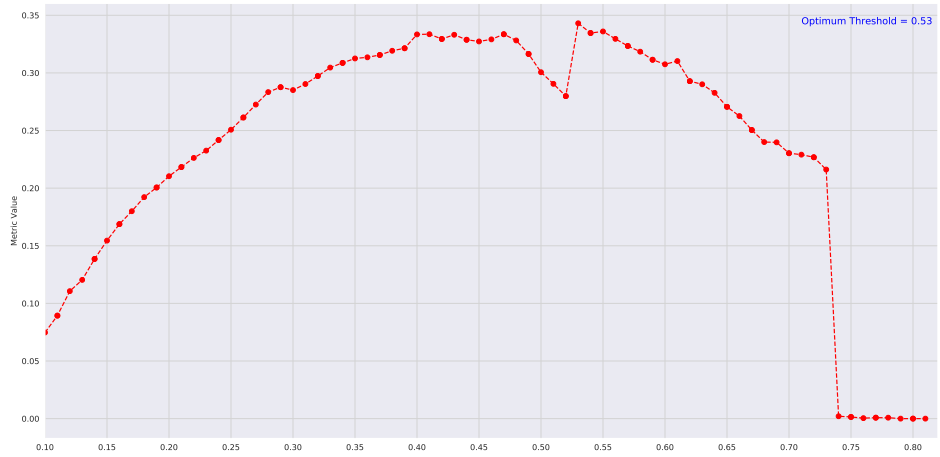
retain all the structured data. For this approach, we would need to impute all the unstructured data which can either be done by filling in empty strings for missing notes or by forward filling missing notes. However, both of these approaches resulted in very bad performances. When missing notes are replaced with empty strings, it resulted in a very sparse input data source. Furthermore, adding a lot of empty strings as clinical notes changed the distribution of data decreasing the prevalence of the positive class and artificially inflating the importance of the structured data. On the other hand, forward filling clinical notes resulted in very large feature space as clinical notes are dense and packed with information with a 60,000 dimension feature representation.

We believe that unstructured data holds more vital information than structured, thus we were willing to compromise on structured data if we could gather all the clinical notes. In order to accomplish this, we used the time each clinical note was recorded as *pivot* point. In particular, at this time point, we gather the latest value of the vital signs data along with their corresponding change statistics in addition to the clinical note data. In this way, the latest value of the vital signs we get access to is the latest status of the patient. Furthermore, through the change statistics, we get an idea of how the patient medical history has changed over the last 24 hours. Finally, the clinical note provides a doctor's viewpoint of the patient status that may include both history and current status.

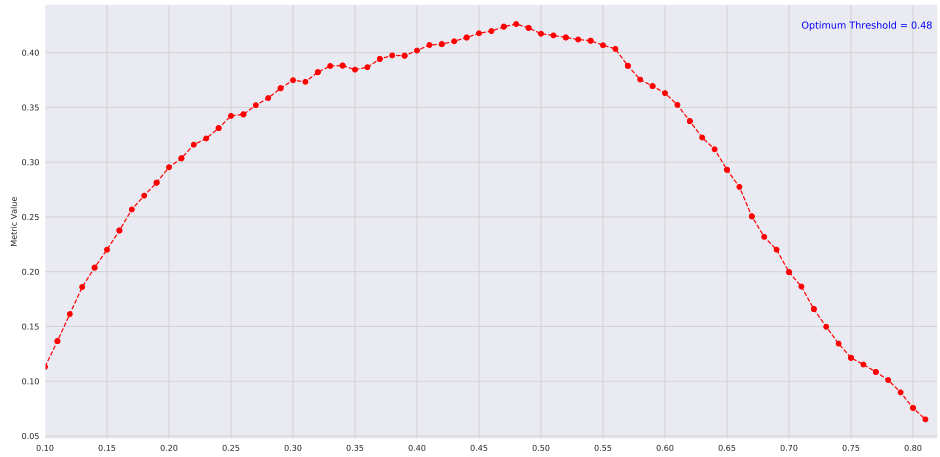
For each encounter, we collect all the data for that encounter and merge them according to the technique described above. This is a novel way of combining structured and unstructured data that are sampled at different frequencies and which have different time intervals between previously recorded data. By integrating both these two types of data into an input source, we are able to build ML models that can utilize the maximum amount of information about an encounter to classifier whether the patient's status has deteriorated to such an extent that the patient has to be admitted to the ICU.

5.6 Model Development

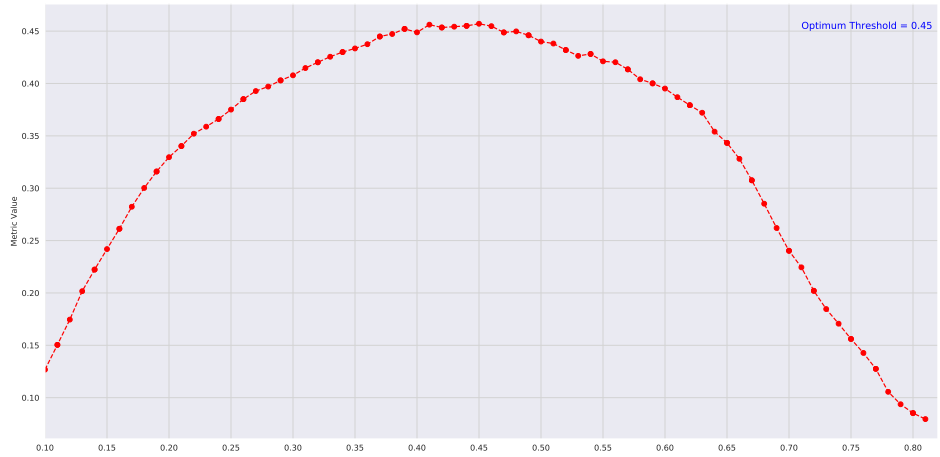
We built 3 models for this task: logistic regression (LR), random forests (RF), and gradient boosting machines (GBM). We used the algorithms for building the models and for evaluating the models iterating over 100 splits between training and test sets. For the model parameters please see appendix B.



(a)Structured Data

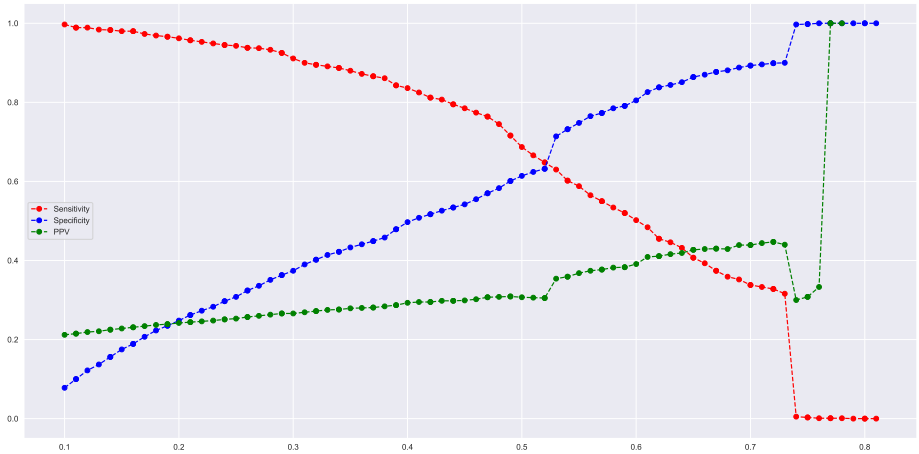


(b)Unstructured Data

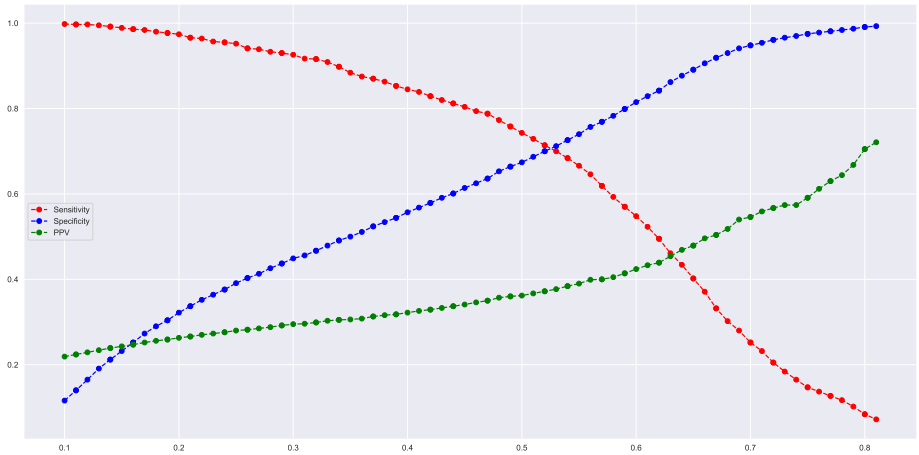


(c)Multimodal Data

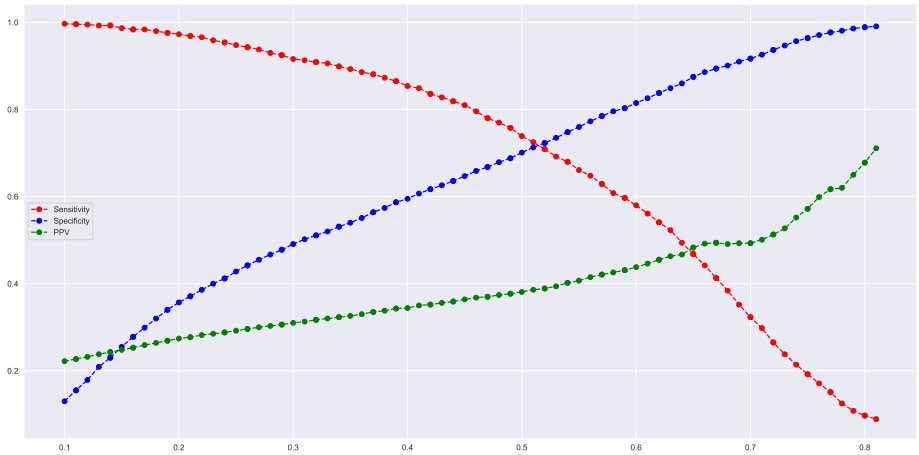
Figure 5-8. *Youden* Index variation for gradient boosting machines model across discrimination thresholds



(a)Structured Data



(b)Unstructured Data



(c)Multimodal Data

Figure 5-9. Performance metrics variation for gradient boosting machines model across discrimination thresholds

We used the *Youden* index to select an optimal discrimination threshold, for all the models. As part of selecting this threshold we plotted the value of the *Youden* index as the threshold was varied from 0 to 1 for each model and each data subset. As an example, figures 5-8 and 5-9 show how the *Youden* index and the sensitivity, specificity, and PPV vary as the threshold changes from 0 to 1 for all three subsets of data using the gradient boosting machine (GBM) model. We can see that there is some instability for higher thresholds when using only structured data, while when using either unstructured or multimodal data, the plots are similar. This indicates that the model seems to extract more discriminatory information from the unstructured data. Please see appendix B for figures indicating the variation of *Youden* index and metrics for all models using all subsets of data.

In addition to the 3 models described above, we also built prediction ensembles of combinations of the three models where we aggregate the predictions made by the three models using an ensembling strategy. In particular, we use two different methods of ensembling: average and maximum. For average ensembling, the probabilities of the positive class predicted by each model, along with their discrimination threshold are averaged and used to get the final predictions. Similarly, for maximum ensembling, we take the maximum of the probability by each model, along with the maximum discrimination threshold and use it to get the final predictions. Average ensembling is done for each pair of models while maximum ensembling is done across all 3 models.

5.7 Results

The prevalence of the positive class in this dataset is 24.7%. The prevalence was maintained during splitting the data in a training set and a testing set.

Figures 5-10, 5-11, 5-12, and 5-13 shows the sensitivity, specificity, PPV, and AUC respectively results for all models over 100 iterations of different partitions of the test set. The figures show the box plot for using structured, unstructured, and multimodal data.

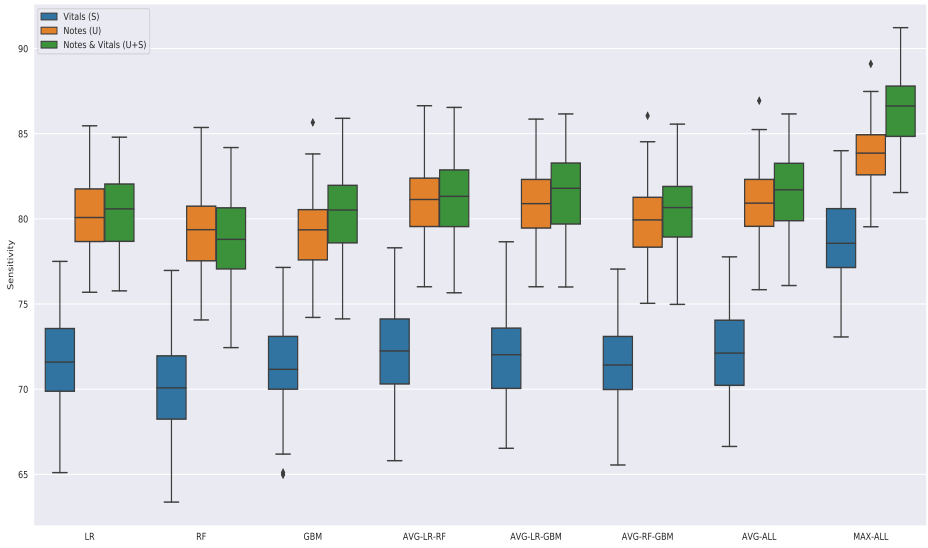


Figure 5-10. Sensitivity Results of all models using different subsets of data

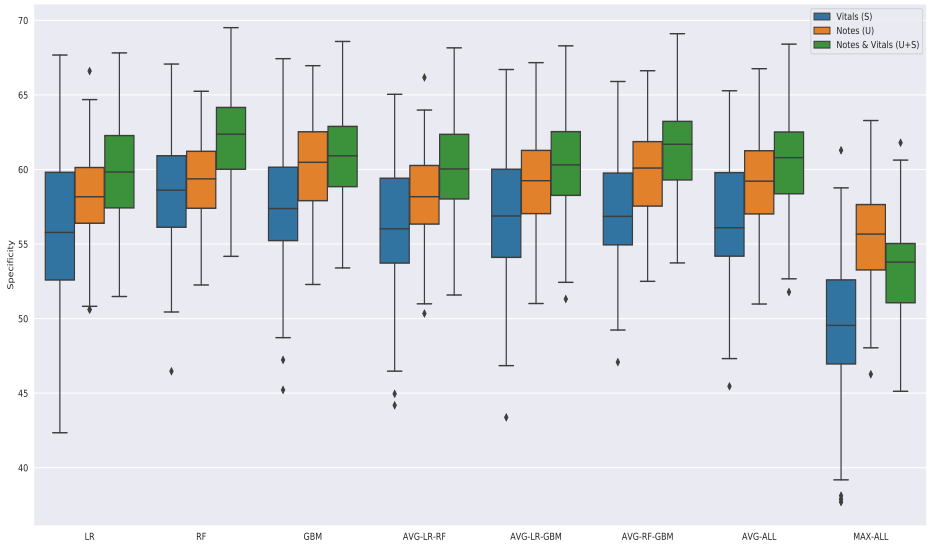


Figure 5-11. Specificity Results of all models using different subsets of data

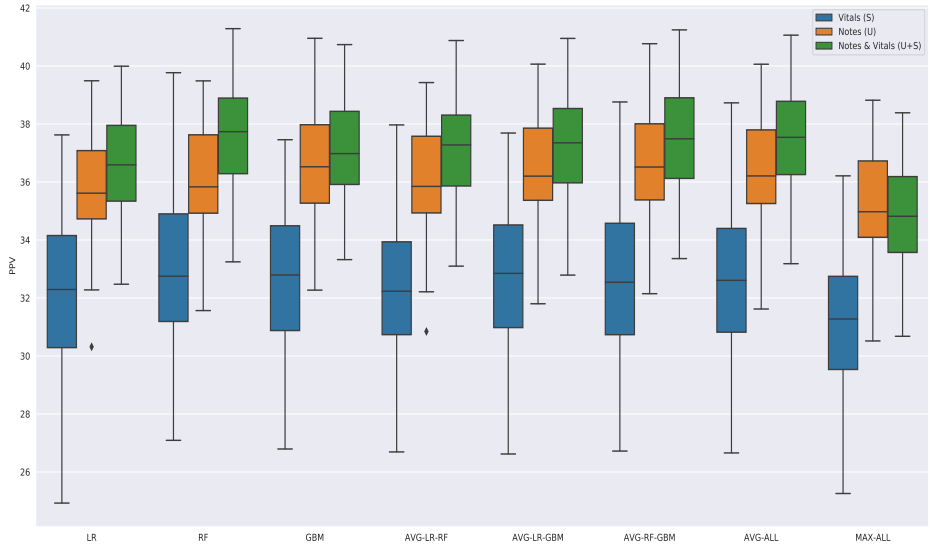


Figure 5-12. PPV Results of all models using different subsets of data

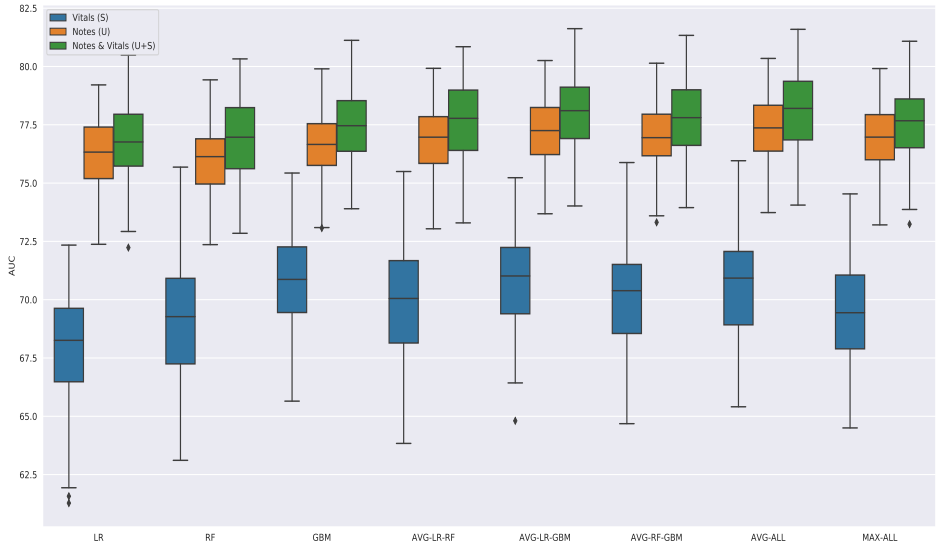


Figure 5-13. AUC Results of all models using different subsets of data

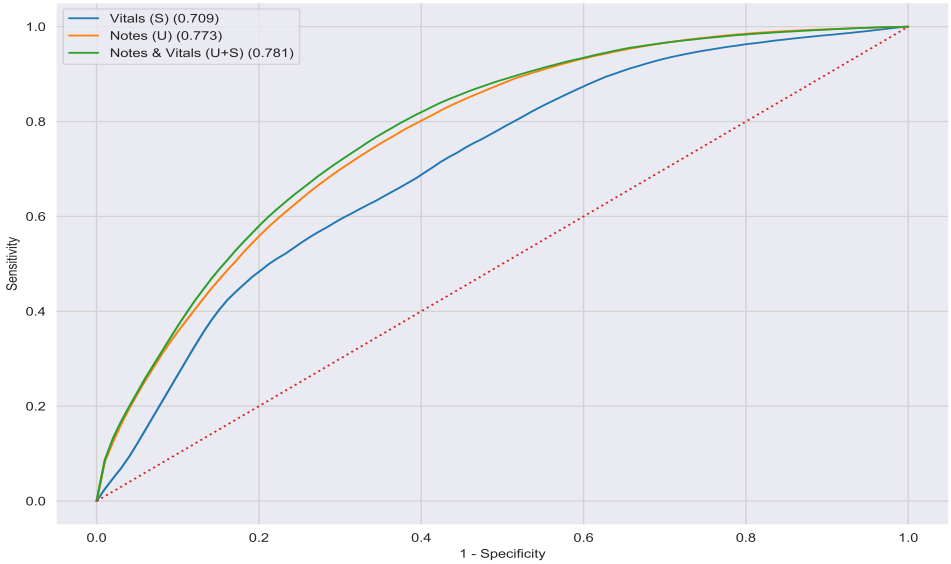


Figure 5-14. Mean ROC curve for the best models

Figure 5-14, shows the mean ROC curve of the best performing model over 100 trials using all modes of data. The mean ROC curves for all models using all modes of data is shown in appendix B. Please also refer to the appendix for the mean confusion matrices for this task for all the models.

Figures 5-15 and 5-16 show the feature importances of the structured and unstructured data respectively. This is an informal way of peaking into what the model thinks are important features in determining imminent ICU admission. Tables 5-3, 5-4, and 5-5 shows the performance metrics of all the models over 100 iterations using structured, unstructured, and multimodal data respectively.

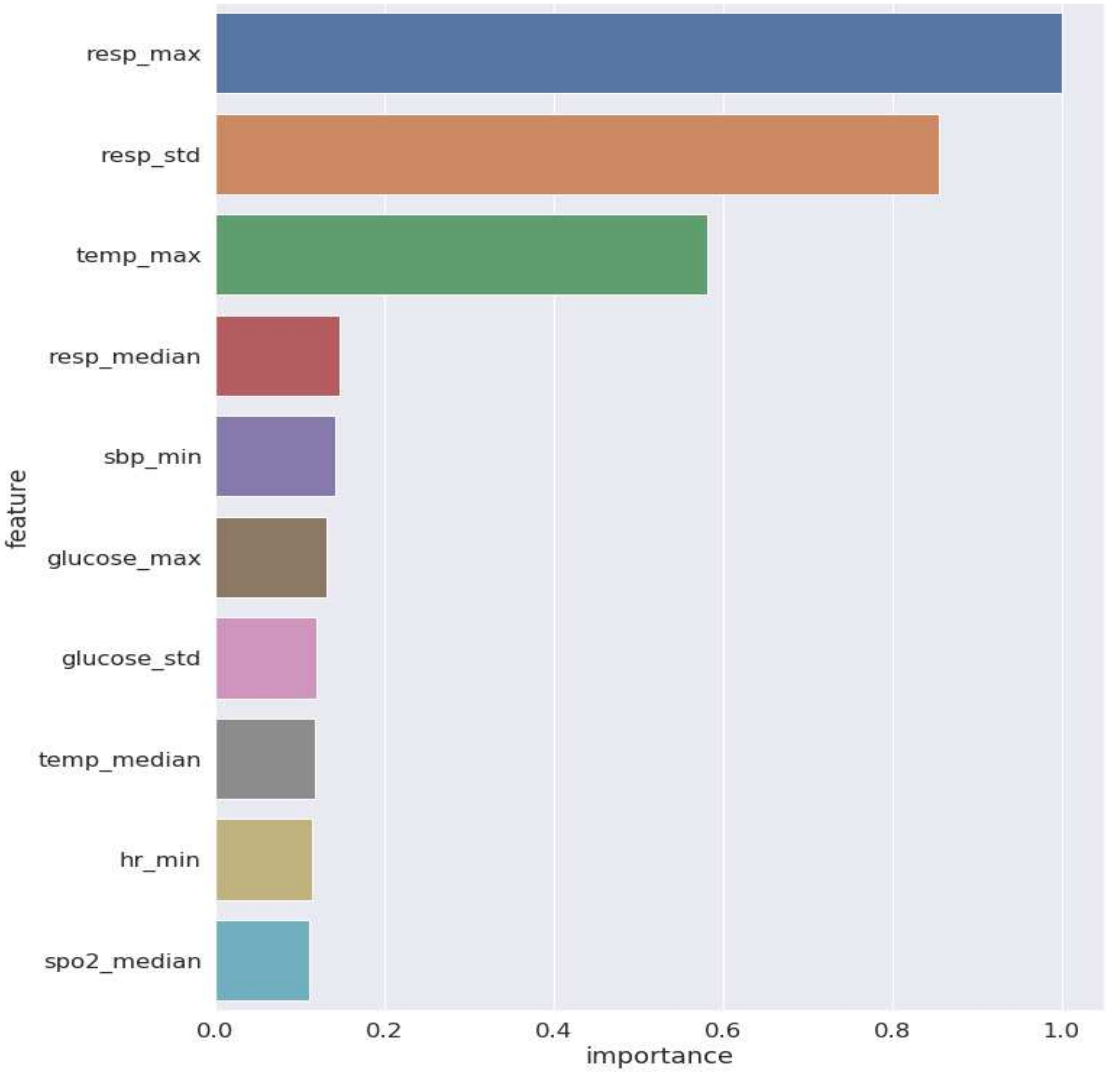


Figure 5-15. Feature importance relative to the most important feature

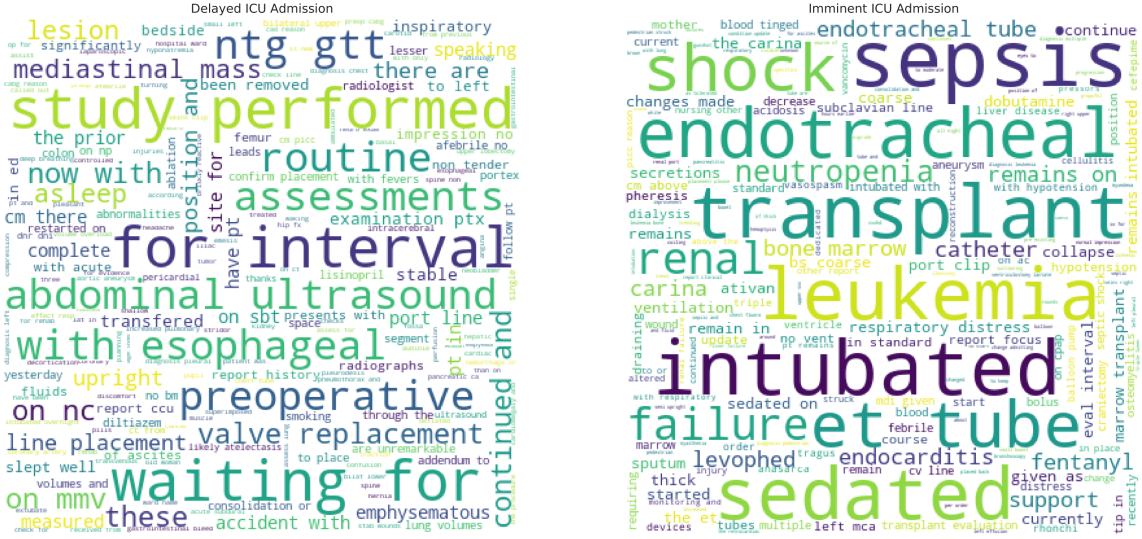


Figure 5-16. Word Cloud of the top 500 tokens produced by the model

Table 5-3. Performance results of models using structured data

Metrics (95% CI)	Sensitivity	Specificity	PPV	AUC
LR	71.6 (71.1 - 72.2)	55.8 (54.9 - 56.8)	32.1 (31.5 - 32.6)	67.9 (67.4 - 68.4)
RF	70.1 (69.6 - 70.6)	**58.5 (57.8 - 59.3)**	**32.9 (32.4 - 33.5)**	69.2 (68.7 - 69.7)
GBM	71.2 (70.7 - 71.7)	57.4 (56.6 - 58.3)	32.7 (32.2 - 33.2)	**70.8 (70.4 - 71.2)**
AVG-LR-RF	72.1 (71.6 - 72.6)	56.1 (55.3 - 56.9)	32.3 (31.8 - 32.8)	69.9 (69.4 - 70.4)
AVG-LR-GBM	71.9 (71.3 - 72.4)	56.9 (56.0 - 57.8)	32.6 (32.1 - 33.2)	70.9 (70.5 - 71.3)
AVG-RF-GBM	71.5 (71.0 - 72.0)	57.1 (56.4 - 57.9)	32.7 (32.1 - 33.2)	70.2 (69.8 - 70.7)
AVG-ALL	72.3 (71.8 - 72.8)	56.5 (55.7 - 57.3)	32.6 (32.0 - 33.1)	70.6 (70.2 - 71.1)
MAX-ALL	**78.8 (78.36 - 79.3)**	49.5 (48.5 - 50.4)	31.2 (30.7 - 31.7)	39.4 (69.0 - 69.9)

Table 5-4. Performance results of models using unstructured data

Metrics (95% CI)	Sensitivity	Specificity	PPV	AUC
LR	80.2 (79.8 - 80.6)	58.3 (57.6 - 58.9)	35.8 (35.4 - 36.1)	76.3 (76.0 - 76.6)
RF	79.1 (78.7 - 79.6)	59.4 (58.8 - 59.9)	36.1 (35.7 - 36.4)	76.0 (75.7 - 76.3)
GBM	79.2 (78.8 - 79.6)	**60.2 (59.6 - 60.8)**	**36.6 (36.2 - 36.9)**	76.6 (76.3 - 77.0)
AVG-LR-RF	81.0 (80.6 - 81.4)	58.4 (57.8 - 59.0)	36.1 (35.7 - 36.4)	76.9 (76.6 - 77.2)
AVG-LR-GBM	80.9 (80.5 - 81.3)	59.2 (58.6 - 59.8)	36.5 (36.1 - 36.8)	77.2 (76.9 - 77.6)
AVG-RF-GBM	79.8 (79.5 - 80.2)	60.0 (59.4 - 60.6)	**36.6 (36.3 - 37.0)**	76.9 (76.6 - 77.2)
AVG-ALL	81.0 (80.6 - 81.4)	59.1 (58.5 - 59.7)	36.4 (36.1 - 36.8)	**77.3 (77.0 - 77.6)**
MAX-ALL	**83.7 (83.3 - 84.1)**	55.4 (54.8 - 56.1)	35.2 (34.9 - 35.6)	76.9 (76.6 - 77.3)

Table 5-5. Performance results of models using multimodal data

Metrics (95% CI)	Sensitivity	Specificity	PPV	AUC
LR	80.3 (79.9 - 80.8)	59.7 (59.0 - 60.4)	36.6 (36.3 - 37.0)	76.8 (76.4 - 77.1)
RF	78.7 (78.3 - 79.2)	**62.1 (61.6 - 62.7)**	**37.6 (37.3 - 38.0)**	77.0 (76.6 - 77.3)
GBM	80.2 (79.8 - 80.7)	60.7 (60.1 - 61.3)	37.2 (36.8 - 37.5)	77.5 (77.1 - 77.8)
AVG-LR-RF	81.2 (80.8 - 81.6)	60.2 (59.5 - 60.8)	37.1 (36.8 - 37.5)	77.7 (77.3 - 78.0)
AVG-LR-GBM	81.5 (81.1 - 82.0)	60.2 (59.5 - 60.9)	37.3 (36.9 - 37.6)	78.0 (77.7 - 77.6)
AVG-RF-GBM	80.4 (80.0 - 80.9)	61.3 (60.7 - 61.9)	**37.6 (37.2 - 37.9)**	77.8 (77.4 - 78.1)
AVG-ALL	81.6 (81.1 - 82.0)	60.5 (59.9 - 61.2)	37.5 (37.1 - 37.8)	**78.1 (77.8 - 78.4)**
MAX-ALL	**86.4 (86.0 - 86.8)**	53.2 (52.5 - 53.9)	34.9 (34.5 - 35.2)	77.6 (77.2 - 77.9)

5.8 Discussion

Our objective is to integrate multimodal data that consists of both structured and unstructured data and build machine learning models and evaluate them to see how augmenting unstructured data with structured data helps or hinders the model's performance. The results provided in the previous section give us some interesting insights.

The first thing we notice is that the models that solely use only structured data perform worse than those that use unstructured data or multimodal data across all metrics. This could be due to several reasons. First, in our dataset, we noticed that a large percentage of the structured data was recorded *after* the last note of a particular encounter. Since we are using the note record time as a pivot point and discarding data taken after the last note, we ended up with those structured data points. Second, there are a total of 51 structured variables enabling 51 dimension vector compared to the 60000 dimension vector representing a single clinical note. Due to this, the unstructured data has more rich and dense information that can be exploited by the model. Finally, currently, we only use the vital signs as part of the structured data and do not include lab results and other types of structured EHR data that could contain more information indicating patient status. This could limit the scope of the contribution of structured data.

Most gains in performance are realized when using unstructured data as input. The best performing models using unstructured data have better performances across all metrics. In particular, we can see significant increases of 6.21% in sensitivity, 2.9% in specificity, 11.25% in PPV, and 9.18% in AUC. This can be attributed to the richness of the information contained in the clinical notes. We noticed this same phenomenon in the previous chapter when we were using only item descriptions for consumer items.

Finally, we can see that across all metrics we have increases in the best performing models when using multimodal data. In particular, in figure 5-14, where we can see that using both structured and unstructured data gives us the best AUC results. While performance gains are significant when comparing the models using only structured data to those using multimodal data, the performance gains are only incremental when comparing against models using unstructured data. In particular, we only get performance increases of 3.22% in sensitivity, 3.16% in specificity, 2.73% in PPV, and 1.03% in AUC.

Sensitivity and specificity are characteristics of the task, while PPV is the clinical relevance of the test. More specifically, PPV is dependent on the prevalence of the positive class, while both sensitivity and specificity are independent of the prevalence. A model with high sensitivity is able to correctly predict imminent ICU admission, while a model with high specificity is able to correctly predict delayed ICU admission. A model with high PPV can be considered more trustworthy in its prediction of imminent ICU admission. Finally, a model with a high AUC has a better capability of distinguishing between both the classes. The trade-off between these metrics is determined by the discrimination threshold.

Our results seem to indicate that the structured data only adds marginal value to the model predictions. However, in a tight and complex scenario such as a hospital ICU where patient's lives are at stake, these small improvements may prove important to the outcome of the patient's medical status. We believe that we can improve these performance gains by adding in the lab information and further processing of clinical notes.

In order to understand which features are important to the model, we extracted the most relevant features considered by the models for the task. While this is an informal non-scientific way of interpreting the model's feature importance, it nonetheless offers some interesting insights. Figure 5-15 shows the top 20 features sorted according to their relative importance to the most important feature. We see that both the top two features comprise of change statistics of the respiratory rate namely the maximum value and the standard deviation over the last 24 hours. This is supported by evidence that respiratory rate variability is a useful predictor of the deterioration of patients [109] which is captured by the maximum, standard deviation, and median. Temperature and glucose variation are also significant factors in determining potential ICU admissions.

We also plotted a word cloud of the top 500 important tokens from the clinical notes show in figure 5-16. For the positive class we can see terms such as *sepsis*, *intubated*, and *endotracheal* appearing with higher weights. Again these terms are associated with the respiratory system which we saw earlier with the structured data. Furthermore, most of the terms associated with the positive class have a sense of "urgency" in them. On the other hand, the words associated with the negative class is indicative of a non-critical scenario.

We developed multiple machine learning models to predict an imminent ICU admission in the next 24-48 hours, by integrating structured and unstructured data. Structured data were extracted from electronic health records, while unstructured data consisted of notes written by healthcare professionals. Leveraging standard NLP techniques such as TF-IDF vectorization, the machine learning models were able to implicitly identify and extract clinical terms to yield good performance for imminent ICU admission prediction task. This indicates that clinical notes can be used to build prediction models to predict imminent ICU admission in the next 24-48 hours. Our approach to predicting imminent ICU admission is distinct from previous work, which primarily uses structured data such as severity scores for predicting the admission outcome. While the performance gains might be marginal, the results still reinforce our hypothesis that integrating multimodal data leads to improved model performance.

CHAPTER 6

CASE STUDY: PREDICTING IMMINENT ICU ADMISSION USING TRANSFER LEARNING WITH CLINICAL NOTES

The MIMIC-III dataset consists of patients admitted to critical care units. Due to the nature of the dataset, there is an inherent bias for the imminent ICU admission task, as most of the patients in the dataset eventually end up in the ICU. A question that arises then is, how would a model perform on this task, when the input data consists of patients that do not go to the ICU at all? To answer this question, we needed a different dataset.

Methodist Le Bonheur Healthcare (MLH) is a hospital that is located in Memphis, TN which routinely sees multiple patients with different conditions. This dataset includes a mix of patients who do not get admitted to the ICU and who get admitted to the ICU. Using this dataset, we try to predict an imminent ICU admission of a patient. We call this dataset *MLH datset*. Currently, we only have access to the clinical notes in this dataset and do not have access to the structured data. As such, we want to focus on on-going work on *transfer learning* with unstructured data where we train on a dataset and test on a different dataset. In this chapter, we describe this work and preliminary results.

6.1 Data Setup & Exploration

We follow the same procedure for filtering and labeling the data as specified in the previous chapter. The characteristics of the cohort after filtering by the exclusion criteria are given in table 6-1.

Figure 6-1 shows the distribution of the note lengths in characters. We can see that most of the notes are within 2000 characters. Compared to the 38,112 notes in the MIMIC dataset, the MLH dataset has 116,400 notes.

Figure 6-2 figure shows the distribution of notes as a function of time to ICU admission. The distribution is similar to the one shown in figure 5-5 for the MIMIC

Table 6-1. Characteristics of MLH cohort cohort excluding unused notes

Characteristics	Value
Patients	2,508
Male, No. (%)	1,276 (50.9)
Age, mean (SD) [IQR], y	61.8 (17.5) [54.0 – 74.0]
Time to ICU admission, mean (SD) [IQR], d	17.6 (20.0) [7.4 – 19.2]
Ethnicity, No (%)	
Asian	12 (0.5)
Black or African American	1396 (55.7)
Other/Unknown	67 (2.7)
White	1,023 (40.8)
Average number of clinical notes per encounter	43.0
Clinical Note Length, mean [SD] [IQR]	4,333.4 [6818.0] (667.0 - 6,094.0)

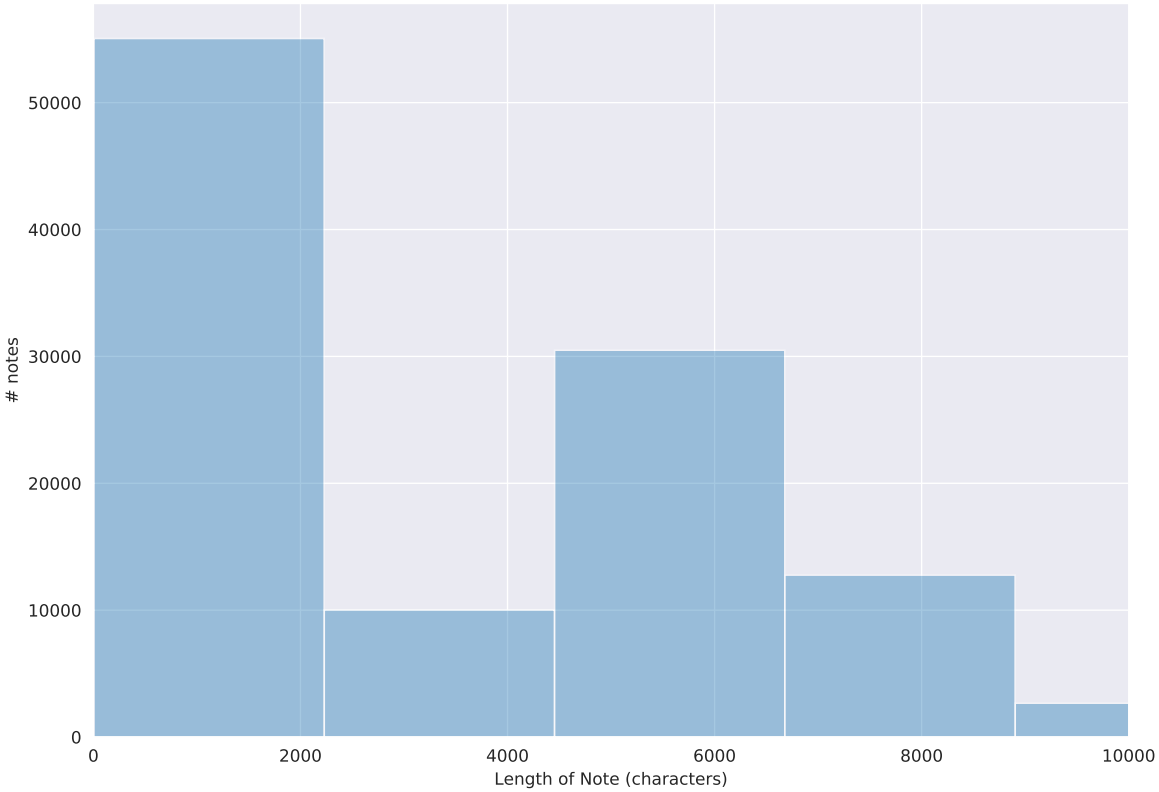


Figure 6-1. Histogram of Note Length

dataset, in that the majority of the notes were recorded within 24 hours before the ICU admission. This gives us the confidence that both the datasets have a similar data distribution.

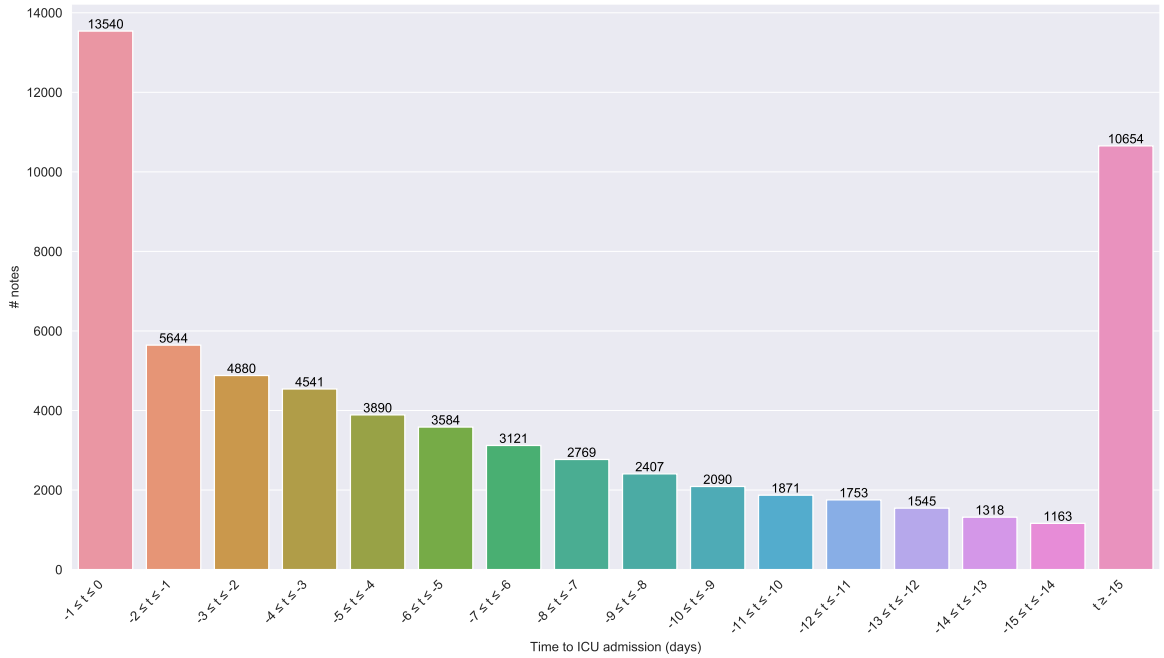


Figure 6-2. Distribution of notes with time to ICU admission

Finally, figure 6-3 shows the histogram of notes by class label. Note that the *Unused* label pertains to those notes that fall outside the boundaries of the time limits we defined. These notes will not be included in building the model. We can see that the dataset is highly imbalanced with the prevalence of the positive class being only 3.4%.

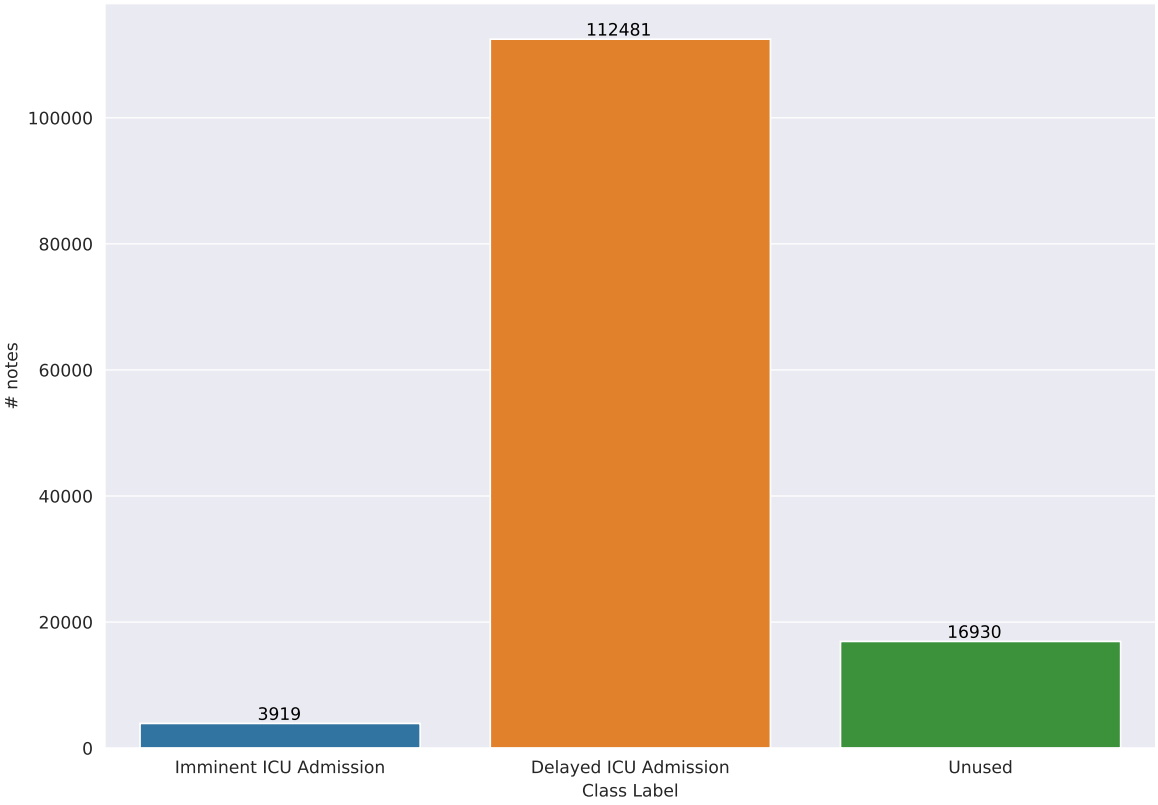


Figure 6-3. Histogram of notes by class label

6.2 Model Development

Our main focus in this chapter is transfer learning and how it can be exploited to get performance from our models. To facilitate that, we perform three different experiments and compare the results. First, we performed an experiment using the clinical notes in the MLH dataset to predict imminent ICU admission. Next, we performed two types of transfer learning: cross-testing and cross-training. Cross-testing refers to training a model on one dataset and testing it without any further training on another dataset, while cross-training refers to training a model on one

dataset and then further training it on a portion of another dataset and testing it on the remaining portion.

For this task, we chose the clinical notes in the MLH dataset as the source data and the clinical notes in the MIMIC dataset as the target data. We chose this setup because of two reasons: 1) the MLH dataset is far bigger than the MIMIC dataset with more than double the number notes; 2) the MLH dataset is more diverse as it contains patients who do not go to the ICU at all. In particular, we chose to train on 10% on the target data for the cross-training stage of the transfer learning experiment.

Similar to previous chapters we build 3 models: logistic regression (LR), random forests (RF), and gradient boosting machines (GBM) and associated ensembles. For how the *Youden* index and performance metrics vary across different discrimination thresholds please see appendix C.

6.3 Results

6.3.1 Imminent ICU admission prediction using only MLH clinical notes

Figures 6-4, 6-5, 6-6, and 6-7 provide a box plot of the sensitivity, specificity, PPV, and AUC results of all models over 100 iterations of the test subset respectively.

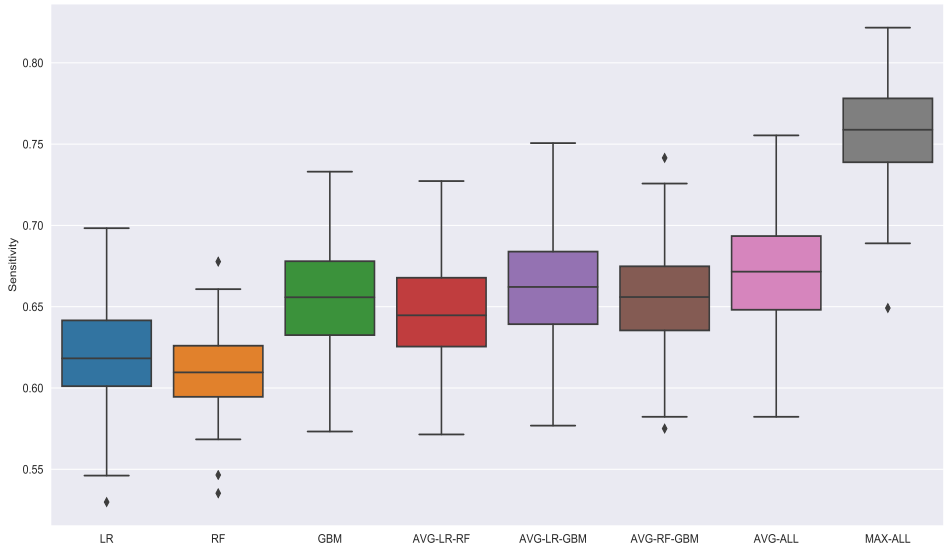


Figure 6-4. Sensitivity Results of all models using clinical notes from MLH dataset

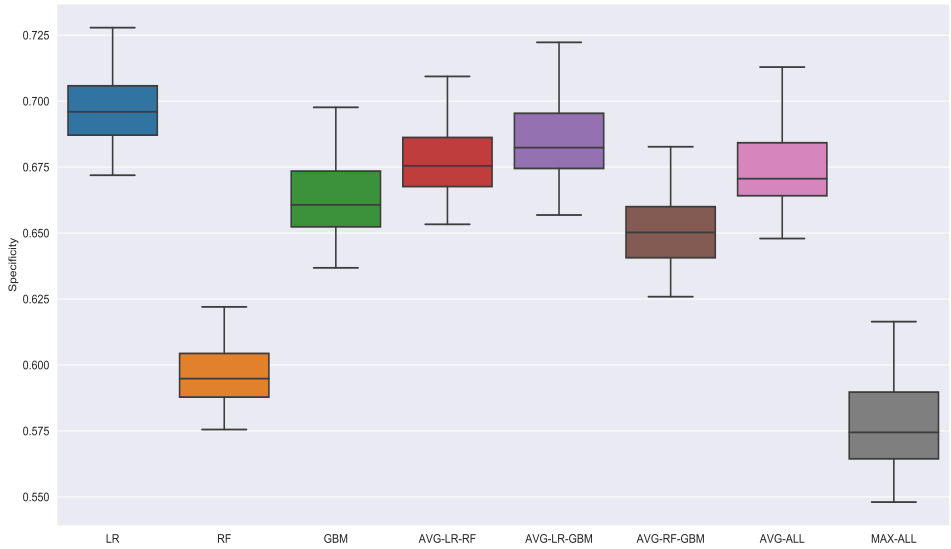


Figure 6-5. Specificity Results of all models using clinical notes from MLH dataset

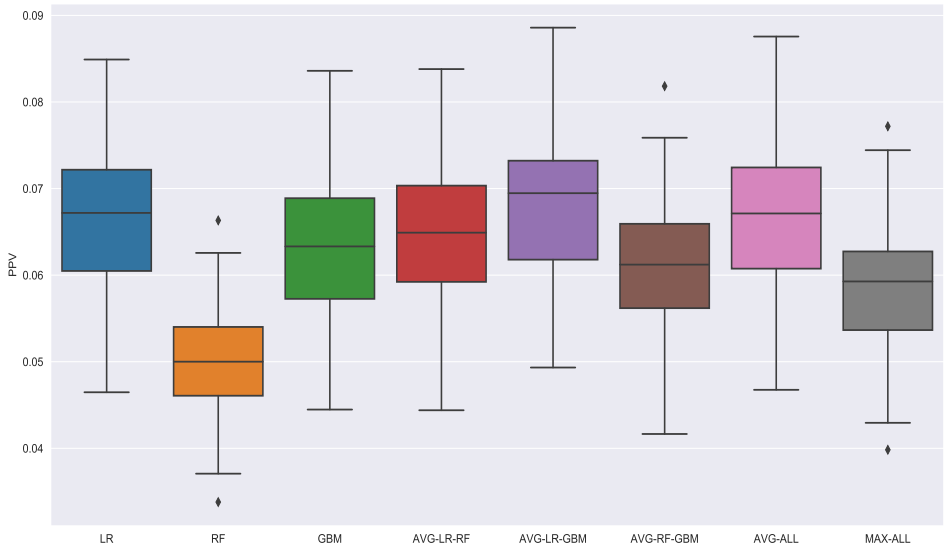


Figure 6-6. PPV Results of all models using clinical notes from MLH dataset

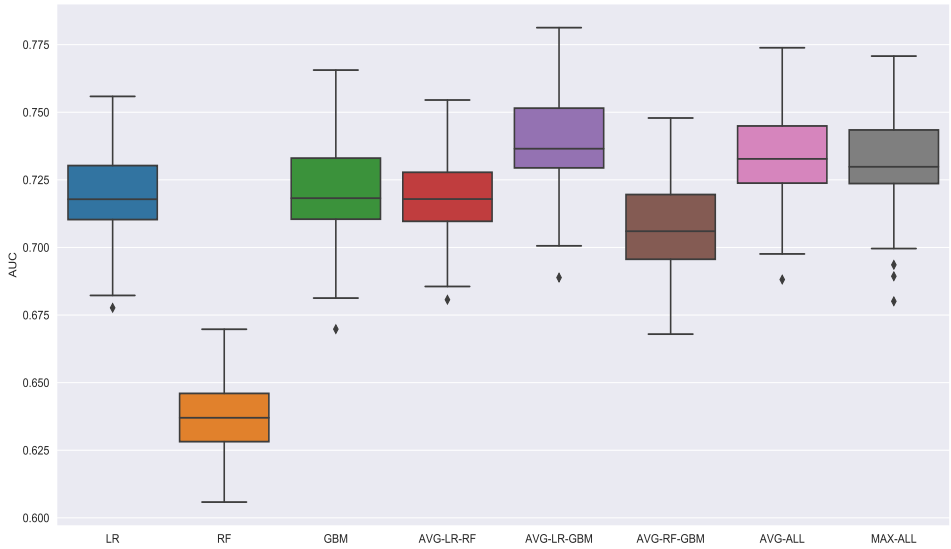


Figure 6-7. AUC Results of all models using clinical notes from MLH dataset

Figure 6-8 shows the mean ROC curve for all models over 100 iterations of the test subset. Similar to the previous chapter, we plotted a word cloud identifying the top 500 tokens that the model considered as important for both the classes. This is shown in figure 6-9.

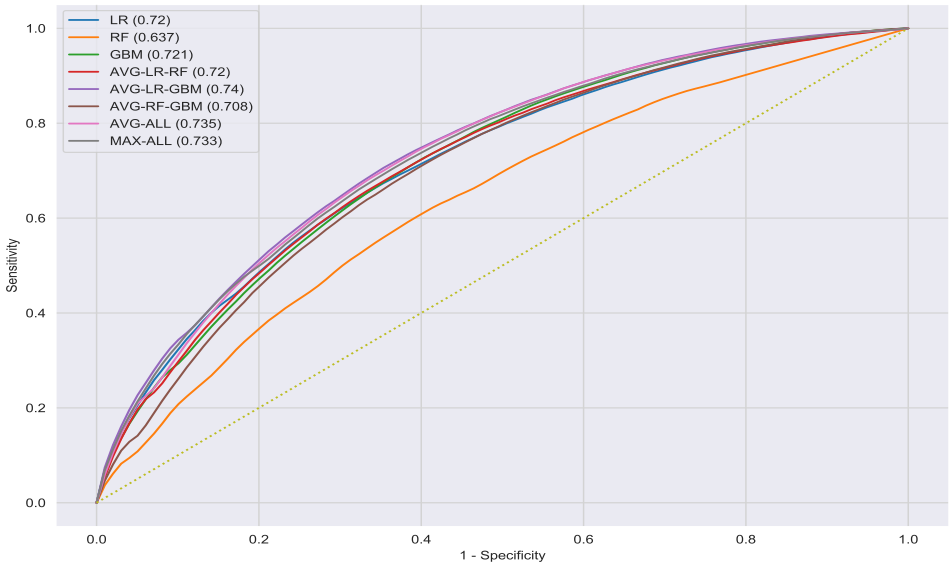


Figure 6-8. Mean ROC curve for all models

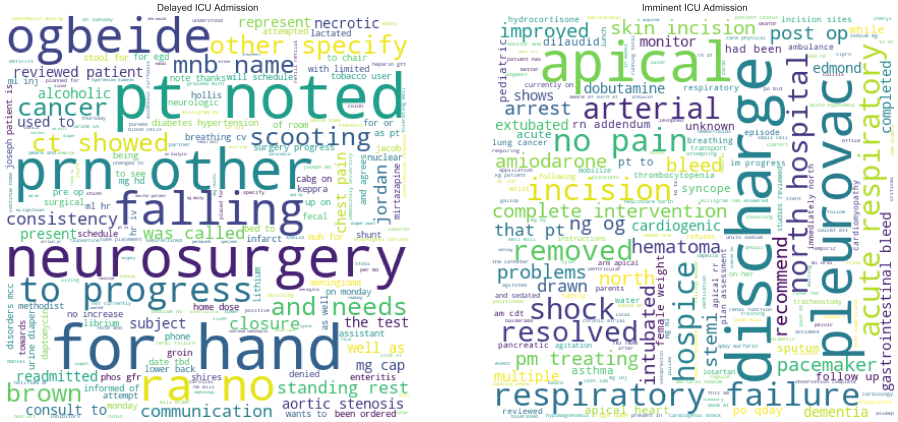


Figure 6-9. Word Cloud of the top 500 tokens produced by the model

Table 6-2. Performance results of models using clinical notes from MLH dataset

Metrics (95% CI)	Sensitivity	Specificity	PPV	AUC
LR	62 (61.4 - 62.7)	**69.7 (69.5 - 70)**	6.7 (6.5 - 6.8)	72 (71.6 - 72.3)
RF	61.2 (60.7 - 61.7)	59.6 (59.4 - 59.8)	5 (4.9 - 5.1)	63.7 (63.4 - 64)
GBM	65.6 (65 - 66.3)	66.3 (66 - 66.5)	6.3 (6.2 - 6.5)	72.1 (71.8 - 72.5)
AVG-LR-RF	64.6 (63.9 - 65.2)	67.7 (67.4 - 68)	6.5 (6.3 - 6.7)	72 (71.6 - 72.3)
AVG-LR-GBM	66.4 (65.7 - 67.1)	68.5 (68.2 - 68.8)	**6.8 (6.7 - 7)**	74 (73.7 - 74.4)
AVG-RF-GBM	65.7 (65.1 - 66.3)	65.1 (64.8 - 65.3)	6.1 (6 - 6.3)	70.8 (70.4 - 71.1)
AVG-ALL	67.1 (66.4 - 67.8)	67.4 (67.1 - 67.6)	6.7 (6.5 - 6.8)	**73.5 (73.1 - 73.8)**
MAX-ALL	**75.8 (75.2 - 76.5)**	57.7 (57.4 - 58)	5.9 (5.7 - 6)	73.3 (72.9 - 73.6)

Table 6-3 shows the performance metrics of the models by cross testing and cross training.

Table 6-3. Performance metrics of models after transfer learning from MLH dataset to MIMIC dataset

Metrics	**Sensitivity**		**Specificity**		**PPV**		**AUC**	
Type of Transfer	**Cross Testing**	**Cross Training**	**Cross Testing**	**Cross Training**	**Cross Testing**	**Cross Training**	**Cross Testing**	**Cross Training**
LR	54.3	71.7 (32.04%)	61.5	**62.9 (2.28%)**	31.6	38.8 (22.78%)	61	74.1 (21.48%)
RF	66.5	72.6 (9.17%)	46.5	61.1 (31.4%)	29	38 (31.03%)	58.1	73.7 (26.85%)
GBM	67.3	71.6 (6.39%)	52.1	61.9 (18.81%)	31.6	38.1 (20.57%)	63.1	73.2 (16.01%)
AVG-LR-RF	57.1	72.9 (27.67%)	58.7	62.2 (5.96%)	31.2	38.8 (24.36%)	61.4	74.4 (21.17%)
AVG-LR-GBM	60.4	72.9 (20.7%)	59.1	62.4 (5.58%)	31.1	38.9 (25.08%)	60.6	74.4 (22.77%)
AVG-RF-GBM	68.2	73.1 (7.18%)	50.9	62 (21.81%)	31.3	38.7 (23.64%)	62.5	74.2 (18.72%)
AVG-ALL	62.2	73.5 (18.17%)	56.9	62.2 (9.31%)	32.1	**39 (21.5%)**	63.2	**74.6 (18.04%)**
MAX-ALL	68.9	**78.2 (13.5%)**	50.5	56.9 (12.67%)	31.4	37.3 (18.79%)	63.2	74.2 (17.41%)

6.4 Discussion

We built the same models from the previous chapter on the new MLH data and measured their performance using the same metrics. Fundamentally, the MLH dataset is different than the MIMIC dataset in three ways. First, it is a much bigger dataset with over double the number of clinical notes compared to the MIMIC dataset. Second, it is more diverse as it includes data of patients who do not go to the ICU. Finally, it is a highly imbalanced dataset with the prevalence of the positive class being only 3.4% compared to over 20% prevalence in the MIMIC dataset.

We notice from table 6-2 that performance across all the metrics and all the models are worse compared to the performances using the MIMIC dataset. These could be attributed to the differences in the datasets. Second, we can see that all the models have very poor PPV with the highest PPV of only 6.8%. This is expected since PPV is dependent on the prevalence of the positive class. In the case of sensitivity, the maximum ensemble has the highest value. The maximum ensemble acts like an *OR* gate, where if any of the individual models have a probability value over the discrimination threshold, the sample is classified as positive. Since all the other models have a very low sensitivity compared to the maximum ensemble, this indicates that each model classifies different samples as positive resulting in a diverse set of models. This is akin to boosting many weak learners to get a strong predictor. Despite the large imbalance, the AUC score is in the lower 70s for all the models. This can bee seen in figure 6-7. Recall that the AUC score is independent of the discrimination threshold and serves as an objective metric to compare model performance. This indicates that all 3 models and their ensembles perform similarly on different datasets.

Similar to the word cloud plotted in 5-16, we plotted a word cloud from the models using the clinical notes in the MLH datasets shown in figure 6-9. This is a non-scientific way of accessing what tokens the model deems important. In the positive class, *respiratory failure*, *pleurovac*, *intubated*, and *apcial* are tokens with high weights. *Pleurovac* is a type of chest drainage system which is used in emergencies. Again we can see that these tokens induce a sense of urgency. However, some tokens do not directly correlate to urgency such as *discharge* and *incision*. Similarly, for the negative class, we have a lot of generic tokens that do

not mean anything out of context. These differences could be attributed to the diverse differences between the MIMIC and MLH datasets.

We also performed cross-testing and cross-training experiments with the clinical notes from MLH dataset as the source and the clinical notes from the MIMIC dataset as the target. These results are shown in table 6-3. Through cross-training, we trained ML models on the entire MLH dataset and tested them on the entire MIMIC dataset. Cross-testing gives us marginal performances which are understandable due to the vase differences in the dataset. However, when we cross-train, where we train our models on the entire MLH dataset and then further train them on only 10% of the MIMIC dataset and test them on the remaining 90%, we get significant improvements. In particular, we can see that we have improved the sensitivity by 13.5%, PPV by 21.5%, and AUC by 18.04%. However, our specificity only increases by 2.28% percent. This is due to the sensitivity specificity trade-off and is dependent on the task. While we have not shown it here, we also performed the same experiments with the MIMIC dataset as the source and the MLH dataset as the target. This did not perform well owing to the differences in the dataset since the MIMIC dataset is both smaller and homogeneous.

The significant improvements achieved by cross-training show us the power of transfer learning where we only train on a very small portion of the target dataset. The results shown here are only using clinical notes from the datasets. We believe that incorporating structured data will yield better model performance. Transfer learning is especially useful in situations where we have limited data or data is protected due to privacy concerns. By training models on a large dataset and training them further on a portion of a different but similar dataset, we can transfer knowledge effectively and help build better ML models.

CHAPTER 7
CONCLUSIONS & FUTURE WORK

Our objective in this research is to build a generalized machine learning pipeline that can integrate multimodal data and build models for various downstream tasks. We showed generalizability by applying our pipeline to different domains. We showed the potential of multimodal data by combining both structured and unstructured data and demonstrated how this led to increased model performance.

We first showed the potential of integrating structured and unstructured in a consumer market environment. We applied our pipeline to predict the potential price of an item given its characteristics through structured variables and user-defined item descriptions which served as unstructured data. We showed by using both structured and unstructured data we were able to get performance improvements over 8%.

We then switched gears to the healthcare domain to showcase the generalizability of the pipeline. We tackled the very important problem of predicting an imminent ICU admission, where we build ML models to determine whether a patient's health has deteriorated to an extent that warrants an ICU admission in the next 24-48 hours. We used two different medical datasets for this task: MIMIC dataset that is publicly available and MLH dataset that is from a private hospital.

We used both structured data in terms of vital signs and unstructured data in terms of clinical notes written by healthcare providers from the MIMIC dataset to build models. The healthcare domain presented new challenges in terms of the complexity of the data. We had data that was taken at different frequencies and had different time intervals between them. We presented a novel approach to integrating structured and unstructured data taken at different time intervals. In particular, we created an integration framework that retained all the information from clinical notes while capturing patient status informed through structured data by using change statistics taken over the last 24 hour period.

We performed extensive experiments using structured data, unstructured data, and multimodal data. We build 3 different models each offering different capabilities from the linear logistic regression to the complex gradient boosting machines. We also built ensemble models that could combine the individual models in a very simple manner. Our results indicated that by integrating both structured and unstructured data we were able to achieve better model performance across a wide range of metrics.

We also showed how transfer learning could assist in situations where we have a dearth of data. We showed two forms of transfer learning: cross-testing and cross-training. In cross-testing, we trained our models on the clinical notes from the MLH dataset and tested them on the clinical notes from the MIMIC dataset. In cross-training, we trained our models using clinical notes from the MLH dataset and then further trained them on a small portion of the MIMIC dataset and tested them on the rest of the MIMIC dataset. We demonstrated significantly improved performance using cross-training across all metrics and models.

We are looking at several avenues for our future work. First and foremost, as soon as we get access to the entire MLH dataset, we want to apply our pipeline to both the structured and unstructured data. This includes both building and evaluating our models only on the MLH datasets and transfer learning using cross-testing and cross-training from the MLH dataset to the MIMIC dataset. Second, we want to add more structured variables as part of our structured data. Currently, we have only vital signs as part of our structured data. However, we believe that adding lab data will improve performance, as it will contain information that is closely associated with the patient's health. Third, we would like to perform more advanced domain-specific processing of our unstructured data. For example, we think that using clinical dictionaries and medical concept extraction will help extract more pertinent information from clinical texts. Finally, we would like to use deep learning methods and models in addition to the classical machine learning models that we have used in this research. Deep learning has shown a lot of promise in NLP. However, their use of multimodal data has been limited. We believe that using deep learning in our framework would improve our task performance.

References

[1] Wullianallur Raghupathi and Viju Raghupathi. "Big data analytics in healthcare: promise and potential". In: *Health information science and systems* 2.1 (2014), p. 3.

[2] Xiaolong Jin, Benjamin W Wah, Xueqi Cheng, et al. "Significance and challenges of big data research". In: *Big Data Research* 2.2 (2015), pp. 59–64.

[3] Ziad Obermeyer and Ezekiel J Emanuel. "Predicting the future—big data, machine learning, and clinical medicine". In: *The New England journal of medicine* 375.13 (2016), p. 1216.

[4] Ruth C Carlos, Charles E Kahn, and Safwan Halabi. "Data science: big data, machine learning, and artificial intelligence". In: *Journal of the American College of Radiology* 15.3 (2018), pp. 497–498.

[5] P Galetsi, K Katsaliaki, and S Kumar. "Values, challenges and future directions of big data analytics in healthcare: A systematic review". In: *Social Science & Medicine* (2019), p. 112533.

[6] Natalia Miloslavskaya, Mikhail Senatorov, Alexander Tolstoy, et al. "Information security maintenance issues for big security-related data". In: *International Conference on Future Internet of Things and Cloud*. IEEE. 2014, pp. 361–366.

[7] Benjamin Shickel, Patrick James Tighe, Azra Bihorac, et al. "Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis". In: *IEEE journal of biomedical and health informatics* 22.5 (2017), pp. 1589–1604.

[8] Andrew L Beam and Isaac S Kohane. "Big data and machine learning in health care". In: *Journal of the American Medical Association* 319.13 (2018), pp. 1317–1318.

[9] Danton S Char, Nigam H Shah, and David Magnus. "Implementing machine learning in health care—addressing ethical challenges". In: *The New England journal of medicine* 378.11 (2018), p. 981.

[10] Qingchen Zhang, Laurence T Yang, Zhikui Chen, et al. "A survey on deep learning for big data". In: *Information Fusion* 42 (2018), pp. 146–157.

[11] Sinno Jialin Pan and Qiang Yang. "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* 22.10 (2009), pp. 1345–1359.

[12] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. "A survey of transfer learning". In: *Journal of Big Data* 3.1 (2016), p. 9.

[13] Chuanqi Tan, Fuchun Sun, Tao Kong, et al. "A survey on deep transfer learning". In: *International Conference on Artificial Neural Networks*. Springer. 2018, pp. 270–279.

[14] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, et al. "A Comprehensive Survey on Transfer Learning". In: *arXiv e-prints*, arXiv:1911.02685 (2014).

[15] *Kaggle - Your Home for Data Science*. https://www.kaggle.com/.

[16] Christoper M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2011.

[17] Andreas C Müller. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, 2016.

[18] Matt Harrison. *Machine learning pocket reference: working with structured data in Python*. OReilly Media, Inc., 2019.

[19] Mark Ryan. *Deep Learning with Structured Data*. O'Reilly Media, 2020.

[20] Guthrie S Birkhead, Michael Klompas, and Nirav R Shah. "Uses of electronic health records for public health surveillance to advance public health". In: *Annual review of public health* 36 (2015), pp. 345–359.

[21] Allan F Simpao, Luis M Ahumada, Jorge A Gálvez, et al. "A review of analytics and clinical informatics in health care". In: *Journal of medical systems* 38.4 (2014), p. 45.

[22] Gunasekaran Manogaran and Daphne Lopez. "A survey of big data architectures and machine learning algorithms in healthcare". In: *International Journal of Biomedical Engineering and Technology* 25.2-4 (2017), pp. 182–211.

[23] Vatsal J Saglani, Bharat S Rawal, V Vijayakumar, et al. "Big Data Technology in Healthcare: A Survey". In: *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE. 2019, pp. 1–5.

[24] Peter B Jensen, Lars J Jensen, and Søren Brunak. "Mining electronic health records: towards better research applications and clinical care". In: *Nature Reviews Genetics* 13.6 (2012), p. 395.

[25] Wencheng Sun, Zhiping Cai, Fang Liu, et al. "A survey of data mining technology on electronic medical records". In: *2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*. IEEE. 2017, pp. 1–6.

[26] Pranjul Yadav, Michael Steinbach, Vipin Kumar, et al. "Mining electronic health records (EHRs): a survey". In: *ACM Computing Surveys (CSUR)* 50.6 (2018), p. 85.

[27] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[28] Riccardo Miotto, Fei Wang, Shuang Wang, et al. "Deep learning for healthcare: review, opportunities and challenges". In: *Briefings in bioinformatics* 19.6 (2017), pp. 1236–1246.

[29] Zachary C Lipton, David C Kale, Charles Elkan, et al. "Learning to diagnose with LSTM recurrent neural networks". In: *arXiv preprint arXiv:1511.03677* (2015).

[30] Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, et al. "DoctorAI: Predicting clinical events via recurrent neural networks". In: *Machine Learning for Healthcare Conference*. 2016, pp. 301–318.

[31] Truyen Tran, Tu Dinh Nguyen, Dinh Phung, et al. "Learning vector representation of medical objects via EMR-driven nonnegative restricted Boltzmann machines (eNRBM)". In: *Journal of biomedical informatics* 54 (2015), pp. 96–105.

[32] Trang Pham, Truyen Tran, Dinh Phung, et al. "Deepcare: A deep dynamic memory model for predictive medicine". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer. 2016, pp. 30–41.

[33] Anand Avati, Kenneth Jung, Stephanie Harman, et al. "Improving palliative care with deep learning". In: *BMC medical informatics and decision making* 18.4 (2018), p. 122.

[34] Lance De Vine, Guido Zuccon, Bevan Koopman, et al. "Medical semantic similarity with a neural language model". In: *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*. ACM. 2014, pp. 1819–1822.

[35] Tomas Mikolov, Ilya Sutskever, Kai Chen, et al. "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*. 2013, pp. 3111–3119.

[36] Youngduck Choi, Chill Yi-I Chiu, and David Sontag. "Learning low-dimensional representations of medical concepts". In: *AMIA Summits on Translational Science Proceedings* 2016 (2016), p. 41.

[37] Andrew L Beam, Benjamin Kompa, Inbar Fried, et al. “Clinical Concept Embeddings Learned from Massive Sources of Multimodal Medical Data”. In: *arXiv preprint arXiv:1804.01486* (2018).

[38] Jeffrey Pennington, Richard Socher, and Christopher Manning. “Glove: Global vectors for word representation”. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[39] Mark Steyvers and Tom Griffiths. “Probabilistic topic models”. In: vol. 427. 7. 2007, pp. 424–440.

[40] Marzyeh Ghassemi, Tristan Naumann, Finale Doshi-Velez, et al. “Unfolding physiological state: Mortality modelling in intensive care units”. In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 75–84.

[41] Mohammed Saeed, Mauricio Villarroel, Andrew T Reisner, et al. “Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): a public-access intensive care unit database”. In: *Critical care medicine* 39.5 (2011), p. 952.

[42] David M Blei, Andrew Y Ng, and Michael I Jordan. “Latent Dirichlet Allocation”. In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[43] William S Noble. “What is a support vector machine?” In: *Nature biotechnology* 24.12 (2006), p. 1565.

[44] Li-wei Lehman, Mohammed Saeed, William Long, et al. “Risk stratification of ICU patients using topic models inferred from unstructured progress notes”. In: *AMIA annual symposium proceedings*. Vol. 2012. American Medical Informatics Association. 2012, p. 505.

[45] Li-wei Lehman, William Long, Mohammed Saeed, et al. “Latent topic discovery of clinical concepts from hospital discharge summaries of a heterogeneous patient cohort”. In: *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE. 2014, pp. 1773–1776.

[46] Adji B Dieng, Chong Wang, Jianfeng Gao, et al. “TopicRNN: A Recurrent Neural Network with Long-Range Semantic Dependency”. In: *arXiv preprint arXiv:1611.01702* (2016).

[47] Paulina Grnarova, Florian Schmidt, Stephanie L Hyland, et al. "Neural document embeddings for intensive care patient mortality prediction". In: *arXiv preprint arXiv:1612.00467* (2016).

[48] Alistair EW Johnson, Tom J Pollard, Lu Shen, et al. "MIMIC-III, a freely accessible critical care database". In: *Scientific data* 3 (2016), p. 160035.

[49] Yohan Jo, Lisa Lee, and Shruti Palaskar. "Combining LSTM and latent topic modeling for mortality prediction". In: *arXiv preprint arXiv:1709.02842* (2017).

[50] Swapna Abhyankar, Dina Demner-Fushman, Fiona M Callaghan, et al. "Combining structured and unstructured data to identify a cohort of ICU patients who received dialysis". In: *Journal of the American Medical Informatics Association* 21.5 (2014), pp. 801–807.

[51] Suchi Saria, Gayle McElvain, Anand K Rajani, et al. "Combining structured and free-text data for automatic coding of patient outcomes". In: *AMIA Annual Symposium Proceedings*. Vol. 2010. American Medical Informatics Association. 2010, p. 712.

[52] Elyne Scheurwegs, Kim Luyckx, Léon Luyten, et al. "Data integration of structured and unstructured sources for assigning clinical codes to patient stays". In: *Journal of the American Medical Informatics Association* 23.e1 (2015), e11–e19.

[53] Tian Bai, Ashis Kumar Chanda, Brian L Egleston, et al. "Joint learning of representations of medical concepts and words from ehr data". In: *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE. 2017, pp. 764–769.

[54] Riccardo Miotto, Li Li, Brian A Kidd, et al. "Deep patient: an unsupervised representation to predict the future of patients from the electronic health records". In: *Scientific reports* 6 (2016), p. 26094.

[55] Hanna M Wallach. "Topic modeling: beyond bag-of-words". In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 977–984.

[56] Robert Fisher, Asim Smailagic, and George Sokos. "Monitoring Health Changes in Congestive Heart Failure Patients Using Wearables and Clinical Data". In: *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, pp. 1061–1064.

[57] Alvin Rajkomar, Eyal Oren, Kai Chen, et al. “Scalable and accurate deep learning with electronic health records”. In: *NPJ Digital Medicine* 1.1 (2018), p. 18.

[58] Alvin Rajkomar, Eyal Oren, Kai Chen, et al. “Scalable and accurate deep learning with electronic health records”. In: *NPJ Digital Medicine* 1.1 (2018), p. 18.

[59] Jingshu Liu, Zachariah Zhang, and Narges Razavian. “Deep EHR: Chronic Disease Prediction Using Medical Notes”. In: *Proceedings of the 3rd Machine Learning for Healthcare Conference*. Vol. 85. Aug. 2018, pp. 440–464.

[60] Keyang Xu, Mike Lam, Jingzhi Pang, et al. “Multimodal Machine Learning for Automated ICD Coding”. In: *Proceedings of the 4th Machine Learning for Healthcare Conference*. Vol. 106. Aug. 2019, pp. 197–215.

[61] Yuqi Si, Jingqi Wang, Hua Xu, et al. “Enhancing Clinical Concept Extraction with Contextual Embedding”. In: *arXiv preprint arXiv:1902.08691* (2019).

[62] Piotr Bojanowski, Edouard Grave, Armand Joulin, et al. “Enriching word vectors with subword information”. In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 135–146.

[63] Armand Joulin, Edouard Grave, Piotr Bojanowski, et al. “Bag of Tricks for Efficient Text Classification”. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. 2017, pp. 427–431.

[64] Matthew E Peters, Mark Neumann, Mohit Iyyer, et al. “Deep contextualized word representations”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2018, pp. 2227–2237.

[65] Jacob Devlin, Ming-Wei Chang, Kenton Lee, et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4171–4186.

[66] Sebastien Dubois, Nathanael Romano, David C Kale, et al. “Effective Representations of Clinical Notes”. In: *arXiv preprint arXiv:1705.07025* (2017).

[67] Min Zeng, Min Li, Zhihui Fei, et al. "Automatic ICD-9 coding via deep transfer learning". In: *Neurocomputing* 324 (2019), pp. 43–50.

[68] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, et al. "An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition". In: *BMC bioinformatics* 16.1 (2015), p. 138.

[69] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. "Introduction to information retrieval". In: *Proceedings of the international communication of association for computing machinery conference*. Vol. 4. 2008.

[70] Ewout W Steyerberg et al. *Clinical prediction models*. Vol. 381. Springer, 2009.

[71] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, et al. "Scikit-learn: Machine learning in Python". In: *Journal of machine learning research* 12 (2011), pp. 2825–2830.

[72] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[73] Leo Breiman. "Bagging predictors". In: *Machine learning* 24.2 (1996), pp. 123–140.

[74] Jerome H Friedman. "Stochastic gradient boosting". In: *Computational statistics & data analysis* 38.4 (2002), pp. 367–378.

[75] Guolin Ke, Qi Meng, Thomas Finley, et al. "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in Neural Information Processing Systems*. 2017, pp. 3146–3154.

[76] Omer Sagi and Lior Rokach. "Ensemble learning: A survey". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8.4 (2018), e1249.

[77] Yu-Chi Ho and David L Pepyne. "Simple explanation of the no-free-lunch theorem and its implications". In: *Journal of optimization theory and applications* 115.3 (2002), pp. 549–570.

[78] Haibo He and Edwardo A Garcia. "Learning from imbalanced data". In: *IEEE Transactions on knowledge and data engineering* 21.9 (2009), pp. 1263–1284.

[79] Qiong Gu, Li Zhu, and Zhihua Cai. "Evaluation measures of the classification performance of imbalanced data sets". In: *International*

symposium on intelligence computation and applications. Springer. 2009, pp. 461–471.

[80] Oluwasanmi O Koyejo, Nagarajan Natarajan, Pradeep K Ravikumar, et al. "Consistent Binary Classification with Generalized Performance Metrics". In: *Advances in Neural Information Processing Systems 27*. 2014, pp. 2744–2752.

[81] Ronen Fluss, David Faraggi, and Benjamin Reiser. "Estimation of the Youden Index and its associated cutoff point". In: *Biometrical Journal: Journal of Mathematical Methods in Biosciences* 47.4 (2005), pp. 458–472.

[82] Farrokh Habibzadeh, Parham Habibzadeh, and Mahboobeh Yadollahie. "On determining the most appropriate test cut-off value: the case of tests with continuous results". In: *Biochemia medica* 26.3 (2016), pp. 297–307.

[83] Anders Kallner. "Formulas". In: *Laboratory Statistics (Second Edition)*. Second Edition. Elsevier, 2018.

[84] Wei-Hsuan Lo-Ciganic, James L Huang, Hao H Zhang, et al. "Evaluation of Machine-Learning Algorithms for Predicting Opioid Overdose Risk Among Medicare Beneficiaries With Opioid Prescriptions". In: *Journal of the American Medical Association* 2.3 (2019), e190968–e190968.

[85] James D Chalmers, Pallavi Mandal, Aran Singanayagam, et al. "Severity assessment tools to guide ICU admission in community-acquired pneumonia: systematic review and meta-analysis". In: *Intensive care medicine* 37.9 (2011), p. 1409.

[86] Rishikesan Kamaleswaran, Carolyn McGregor, and Jennifer Percival. "Service oriented architecture for the integration of clinical and physiological data for real-time event stream processing". In: *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE. 2009, pp. 1667–1670.

[87] Michael P Young, Valerie J Gooder, Karen Mc Bride, et al. "Inpatient transfers to the intensive care unit: delays are associated with increased mortality and morbidity". In: *Journal of general internal medicine* 18.2 (2003), pp. 77–83.

[88] David JP O'Callaghan, Parveen Jayia, Eyston Vaughan-Huxley, et al. "An observational study to determine the effect of delayed admission to the intensive care unit on patient outcome". In: *Critical Care* 16.5 (2012).

[89] Lucienne TQ Cardoso, Cintia MC Grion, Tiemi Matsuo, et al. "Impact of delayed admission to intensive care units on mortality of critically ill patients: a cohort study". In: *Critical care* 15.1 (2011), R28.

[90] Dina Demner-Fushman, Wendy W Chapman, and Clement J McDonald. "What can natural language processing do for clinical decision support?" In: *Journal of biomedical informatics* 42.5 (2009), pp. 760–772.

[91] Jinsung Yoon, Ahmed Alaa, Scott Hu, et al. "ForecastICU: a prognostic decision support system for timely prediction of intensive care unit admission". In: *International Conference on Machine Learning*. 2016, pp. 1680–1689.

[92] Joseph L Nates, Mark Nunnally, Ruth Kleinpell, et al. "ICU admission, discharge, and triage guidelines: a framework to enhance clinical operations, development of institutional policies, and further research". In: *Critical care medicine* 44.8 (2016), pp. 1553–1602.

[93] Luke A Martin, Julie A Kilpatrick, Ragheed Al-Dulaimi, et al. "Predicting ICU readmission among surgical ICU patients: Development and validation of a clinical nomogram". In: *Surgery* 165.2 (2019), pp. 373–380.

[94] Elsa Loekito, James Bailey, Rinaldo Bellomo, et al. "Common laboratory tests predict imminent medical emergency team calls, intensive care unit admission or death in emergency department patients". In: *Emergency Medicine Australasia* 25.2 (2013), pp. 132–139.

[95] Jun-Yu Wang, Yun-Xia Chen, Shu-Bin Guo, et al. "Predictive performance of quick Sepsis-related Organ Failure Assessment for mortality and ICU admission in patients with infection at the ED". In: *The American journal of emergency medicine* 34.9 (2016), pp. 1788–1793.

[96] André Lavoie, Lynne Moore, Natalie LeSage, et al. "The Injury Severity Score or the New Injury Severity Score for predicting intensive care unit admission and hospital length of stay?" In: *Injury* 36.4 (2005), pp. 477–483.

[97] Franco van Wyk, Anahita Khojandi, Akram Mohammed, et al. "A minimal set of physiomarkers in continuous high frequency data streams predict adult sepsis onset earlier". In: *International journal of medical informatics* 122 (2019), pp. 55–62.

[98] Ben J Marafino, W John Boscardin, and R Adams Dudley. "Efficient and sparse feature selection for biomedical text classification via the elastic net:

Application to ICU risk stratification from nursing notes". In: *Journal of biomedical informatics* 54 (2015), pp. 114–120.

[99] Ben J Marafino, Miran Park, Jason M Davies, et al. "Validation of prediction models for critical care outcomes using natural language processing of electronic health record data". In: *JAMA network open* 1.8 (2018), e185097–e185097.

[100] Gary E Weissman, Rebecca A Hubbard, Lyle H Ungar, et al. "Inclusion of unstructured clinical text improves early prediction of death or prolonged ICU stay". In: *Critical care medicine* 46.7 (2018), pp. 1125–1132.

[101] Thanos Gentimis, Alnaser Ala'J, Alex Durante, et al. "Predicting Hospital Length of Stay Using Neural Networks on MIMIC-III Data". In: *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*. IEEE. 2017, pp. 1194–1201.

[102] Sanjay Purushotham, Chuizheng Meng, Zhengping Che, et al. "Benchmarking deep learning models on large healthcare datasets". In: *Journal of biomedical informatics* 83 (2018), pp. 112–134.

[103] Gokul S Krishnan et al. "Evaluating the quality of word representation models for unstructured clinical Text based ICU mortality prediction". In: *Proceedings of the 20th International Conference on Distributed Computing and Networking*. ACM. 2019, pp. 480–485.

[104] Alistair EW Johnson, Tom J Pollard, and Roger G Mark. "Reproducibility in critical care: a mortality prediction case study". In: *Machine Learning for Healthcare Conference*. 2017, pp. 361–376.

[105] Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, et al. "Transfer learning for clinical time series analysis using recurrent neural networks". In: *arXiv preprint arXiv:1807.01705* (2018).

[106] Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, et al. "Multitask learning and benchmarking with clinical time series data". In: *arXiv preprint arXiv:1703.07771* (2017).

[107] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, et al. "Recurrent neural networks for multivariate time series with missing values". In: *Scientific reports* 8.1 (2018), p. 6085.

[108] Alistair EW Johnson, David J Stone, Leo A Celi, et al. "The MIMIC Code Repository: enabling reproducibility in critical care research". In: *Journal of the American Medical Informatics Association* 25.1 (2017), pp. 32–39.

[109] Daniel Garrido, Justin J Assioun, Anahit Keshishyan, et al. "Respiratory Rate Variability as a Prognostic Factor in Hospitalized Patients Transferred to the Intensive Care Unit". In: *Cureus* 10.1 (2018).

APPENDICES

APPENDIX A
MERCARI PRICE SUGGESTION APPENDIX

For the training set, we gather the mean price for each categorical variable, sort it in descending order and give each category a unique ID. Then the value of the corresponding numeric value of the category is the proportion of its occurrence within the dataset. For example, the *brand_ name* categorical variable is encoded as follows:

Listing A.1: Train Data Mean Target Encoding

```
brands = train_df.groupby('brand_name')['price'].mean().sort_values(
    ascending=False).to_frame()
brands['id'] = brands.reset_index().index.values
brand_names = brands.index.values

train_brand_data = brands.loc[train_df['brand_name']]
train_df.loc[:, 'brand_val'] = train_brand_data['id'].values/len(
    brand_names)
```

For the test set, we do the same thing, but we use the brand indices calculated with the training data. Any unseen brands are considered "missing".

Listing A.2: Test Data Mean Target Encoding

```
test_brand_data = brands.loc[test_df['brand_name']]
test_df.loc[:, 'brand_val'] = test_brand_data['id'].values/len(
    brand_names)
```

We built a gradient boosting model using LightGBM. We built two models: 1) one model that used only the item description and 2) that used the item

description and the structured data available about the item. The parameters of the model are as follows:

Listing A.3: GBM Model Parameters

```
params = {
  'num_leaves': 400,
  'learning_rate': 0.05,
  'feature_fraction': 0.9,
  'bagging_fraction': 0.7,
  'bagging_freq': 5,
  'metric': 'rmse',
  'num_threads': 32,
  'max_bin': 32,
  'objective': 'regression',
}
```

We used a 1000 iteration boost round, evaluating against the validation set to induce early stopping if the validation error did not reduce after 10 rounds. For the final model, we just had a 1000 iteration boost round on the entire training data.

Table A-1. RMSLE of the model with the second test dataset as reported by Kaggle

Data	**RMSLE**
Item description only	0.47
Item description & structured data	0.43

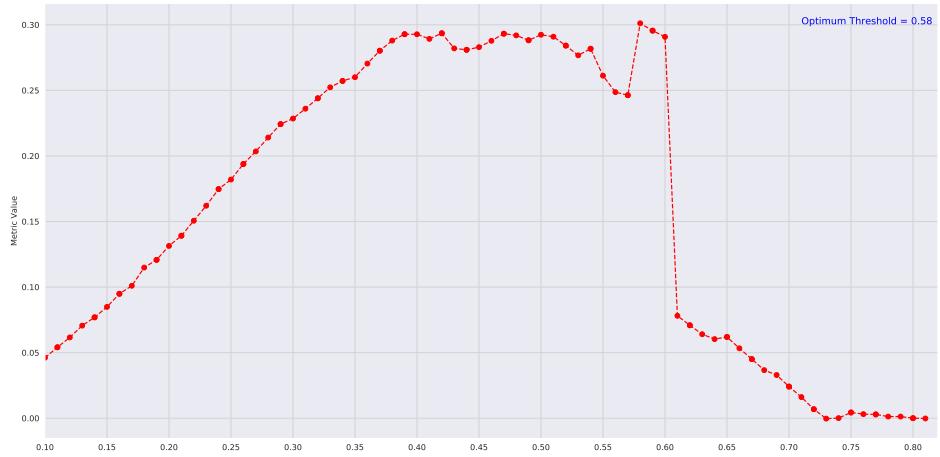
APPENDIX B

PREDICTING IMMINENT ICU ADMISSION USING MIMIC DATASET APPENDIX

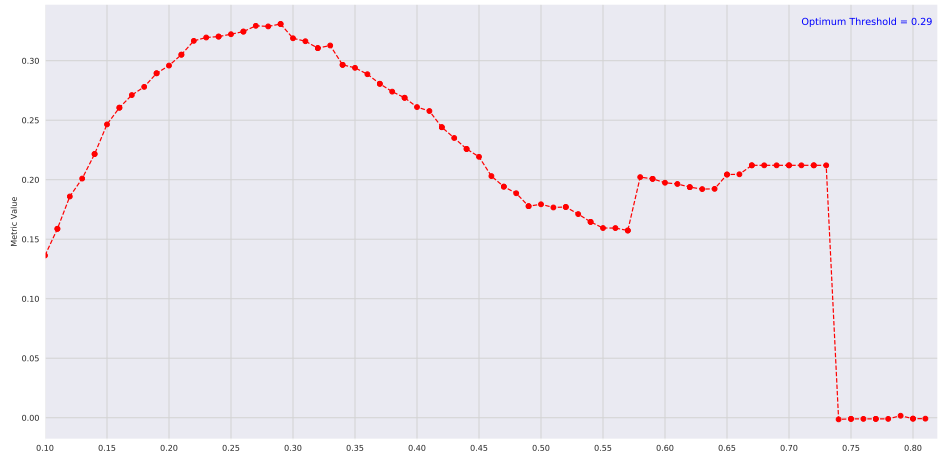
Listing B.1: Model Parameters

```
lr_params = {
  'class_weight': 'balanced',
}

rf_params = {
  'n_estimators': 400,
  'min_samples_leaf': 3,
  'oob_score': True,
  'class_weight': 'balanced',
  'n_jobs': -1,
}

lgb_params = {
  'objective': 'binary',
  'metric': 'binary_logloss',
  'is_unbalance': True,
  'learning_rate': 0.05,
  'max_bin': 16,
  'feature_fraction': 0.5,
}
```

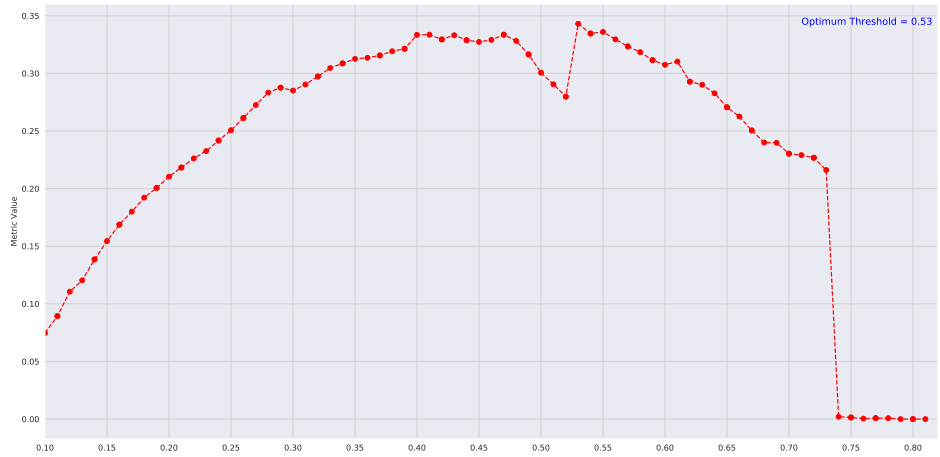
Figure B-1. Distribution of notes with time to ICU admission split by category



(a)Logistic Regression

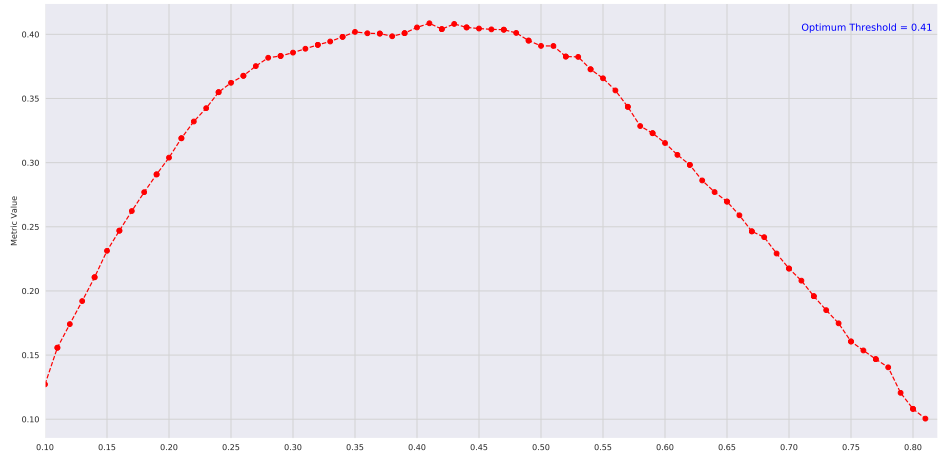


(b)Random Forests

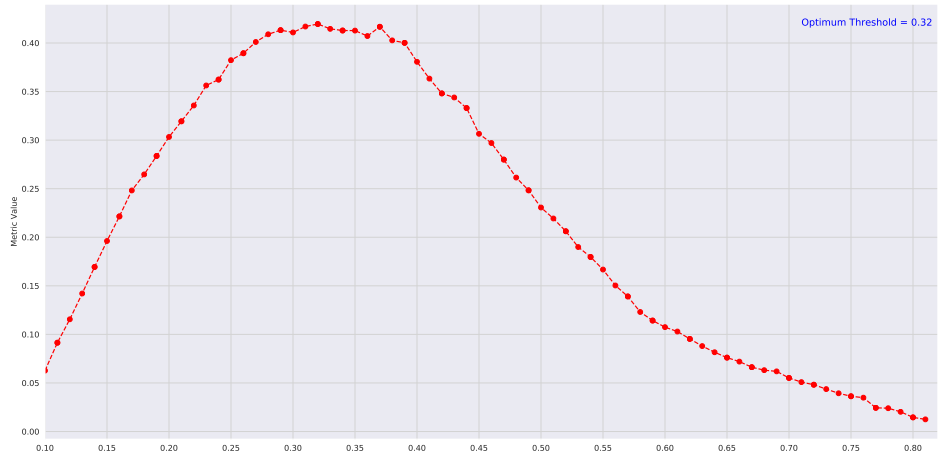


(c)Gradient Boosting Machines

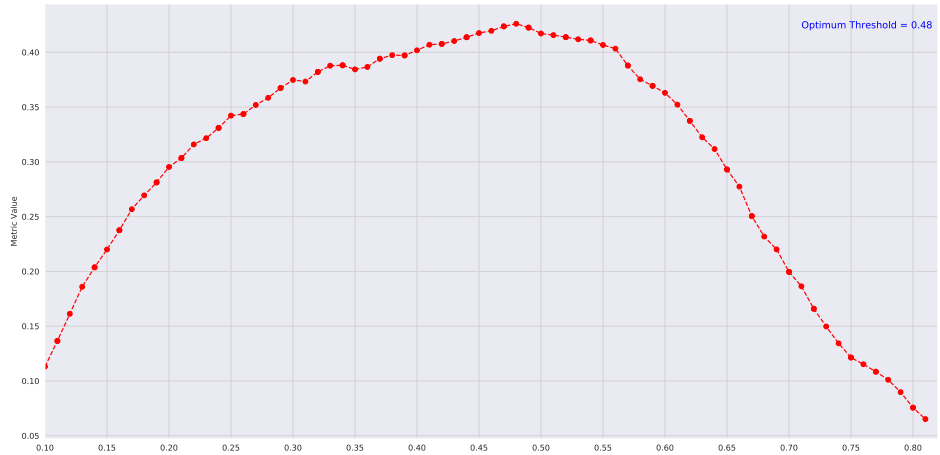
Figure B-2. *Youden* Index variation across discrimination thresholds using structured data



(a)Logistic Regression

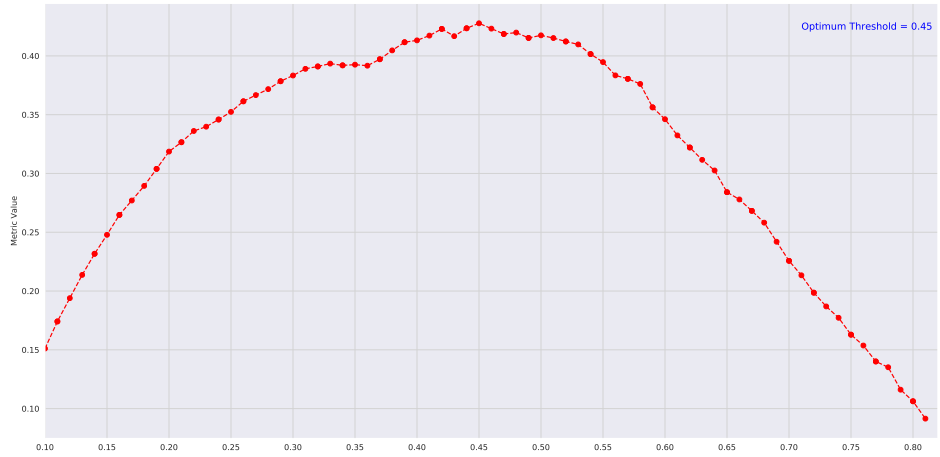


(b)Random Forests

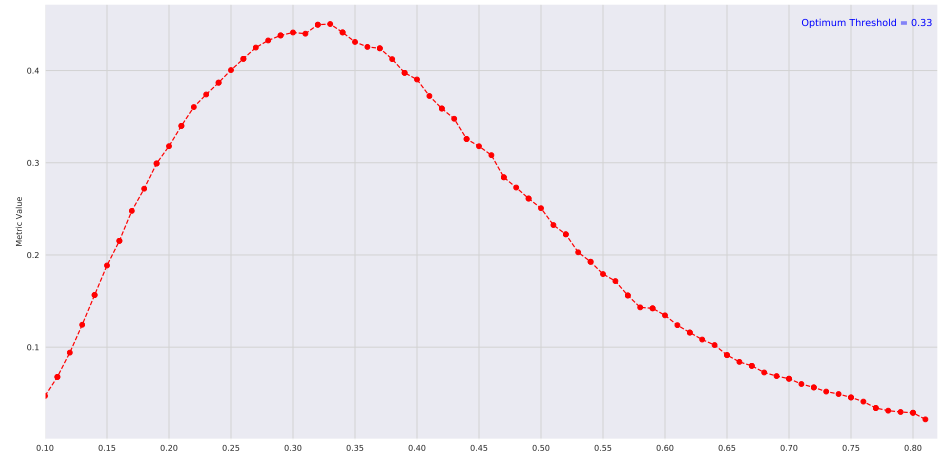


(c)Gradient Boosting Machines

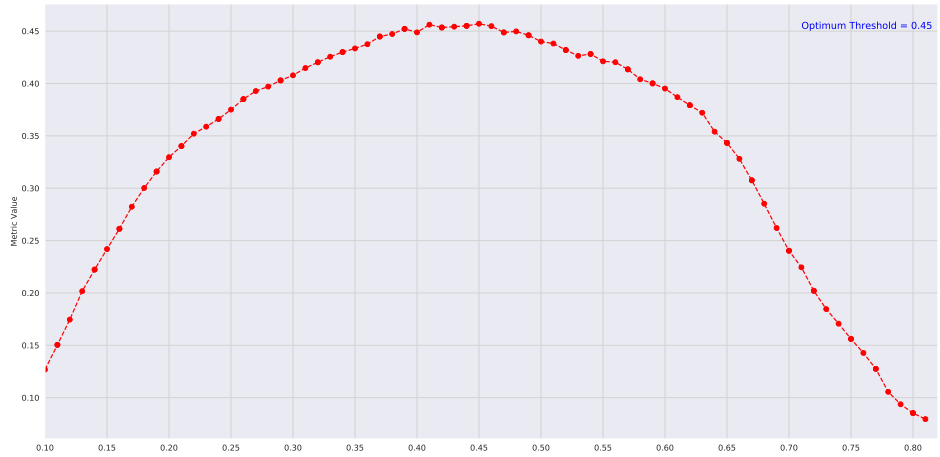
Figure B-3. *Youden* Index variation across discrimination thresholds using unstructured data



(a)Logistic Regression

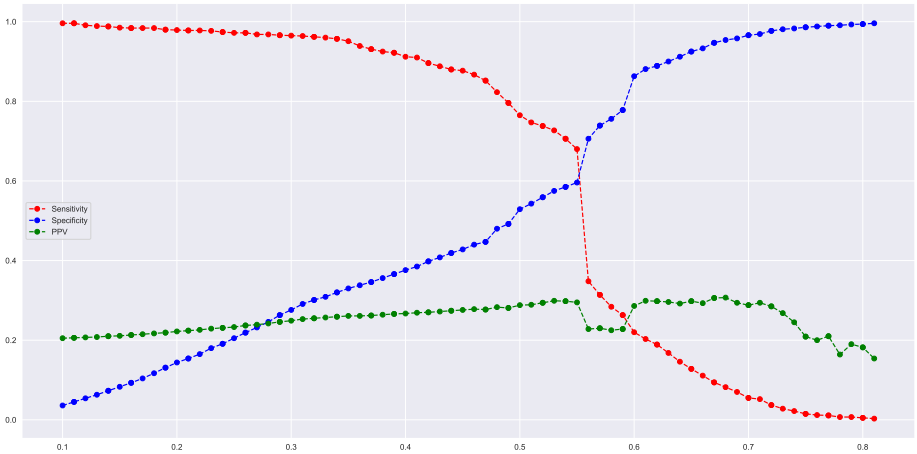


(b)Random Forests

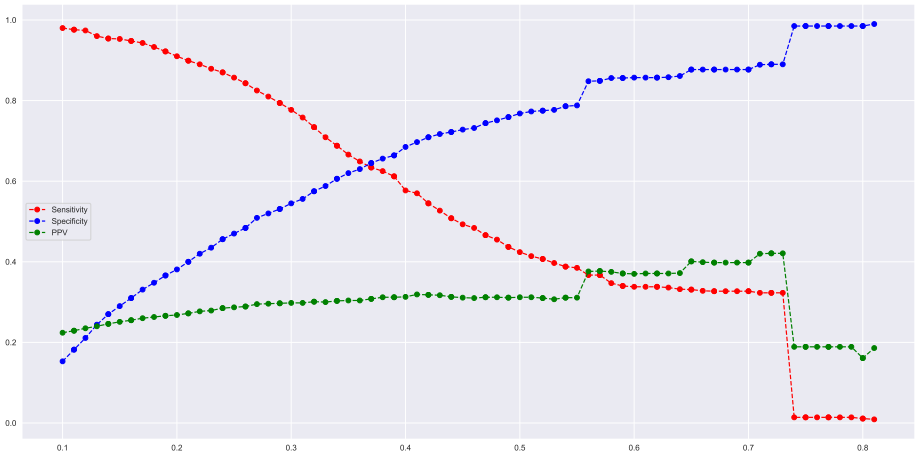


(c)Gradient Boosting Machines

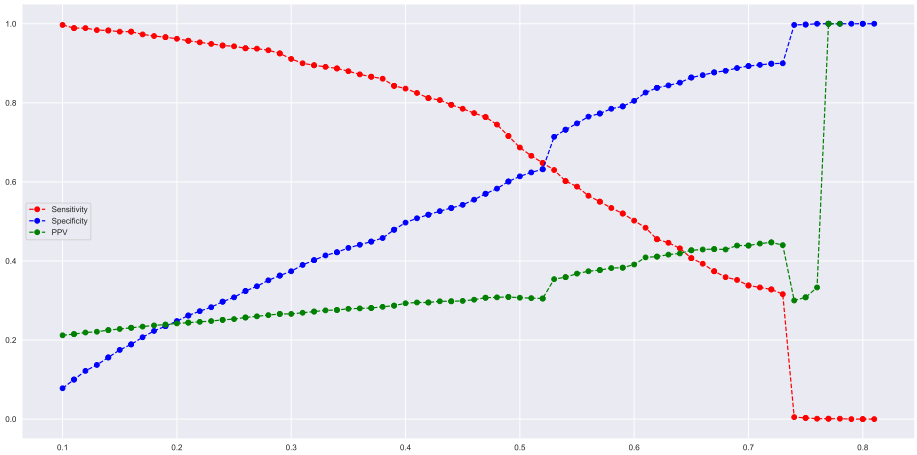
Figure B-4. *Youden* Index variation across discrimination thresholds using multimodal data



(a)Logistic Regression

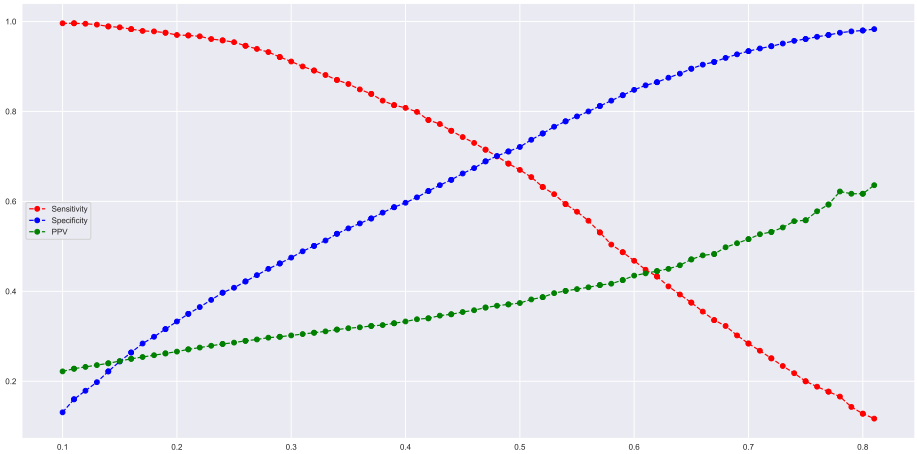


(b)Random Forests

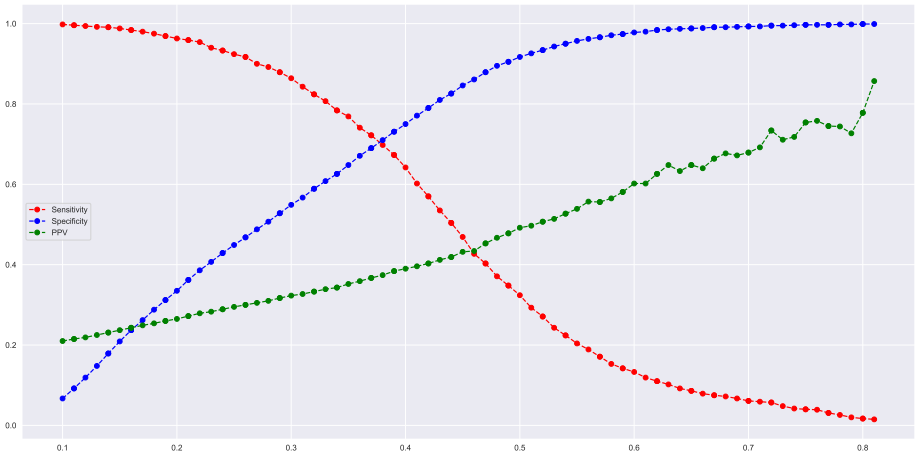


(c)Gradient Boosting Machines

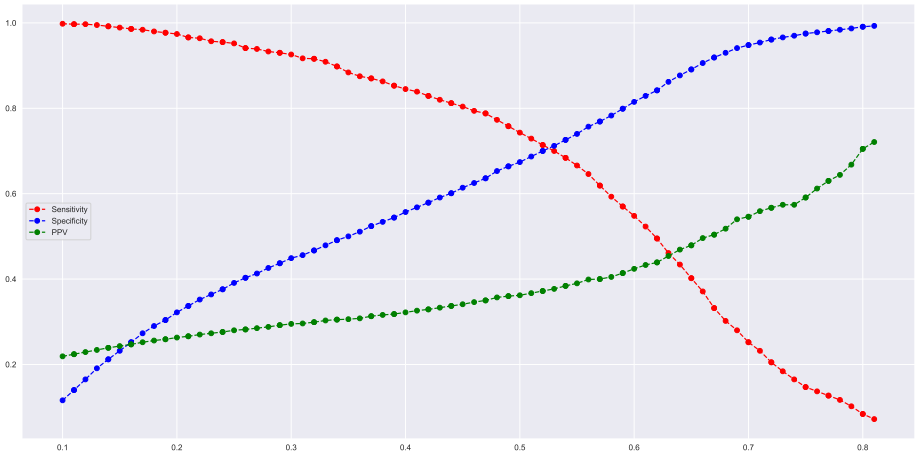
Figure B-5. Performance metrics variation across discrimination thresholds across discrimination thresholds using structured data



(a)Logistic Regression

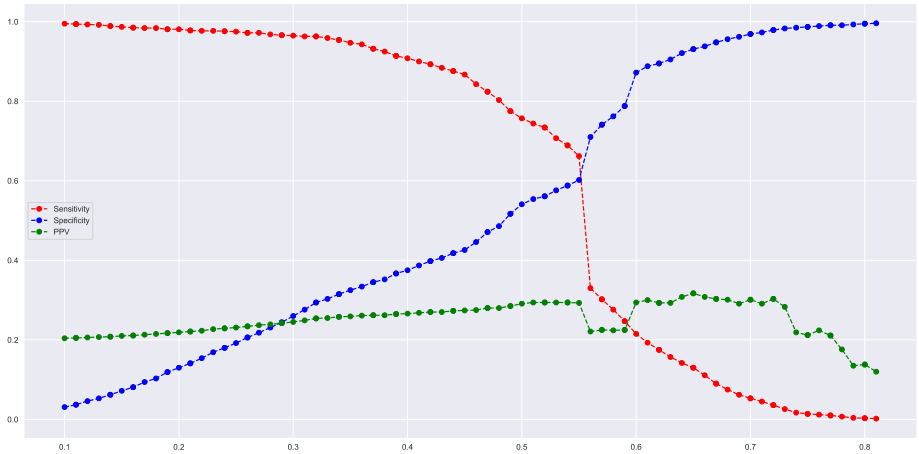


(b)Random Forests

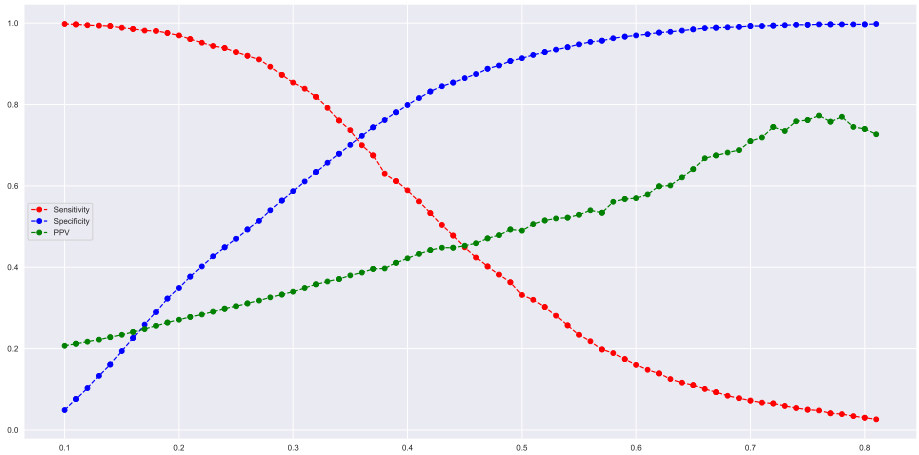


(c)Gradient Boosting Machines

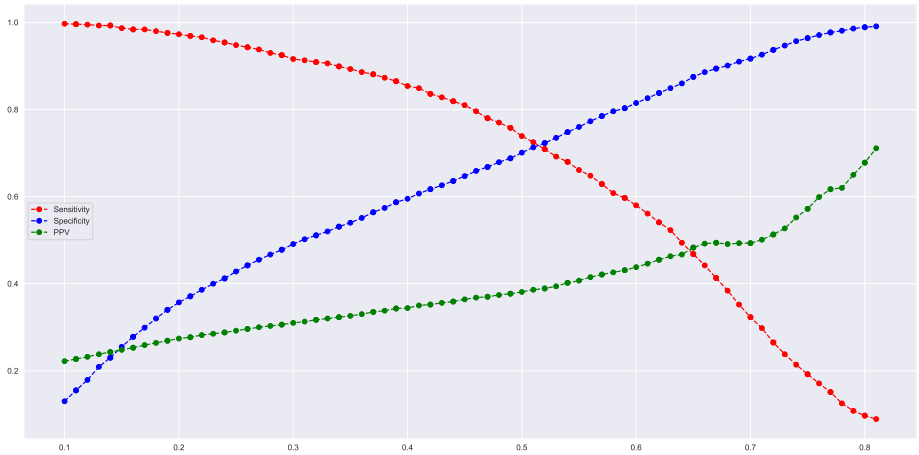
Figure B-6. Performance metrics variation across discrimination thresholds across discrimination thresholds using unstructured data



(a)Logistic Regression



(b)Random Forests



(c)Gradient Boosting Machines

Figure B-7. Performance metrics variation across discrimination thresholds across discrimination thresholds using multimodal data

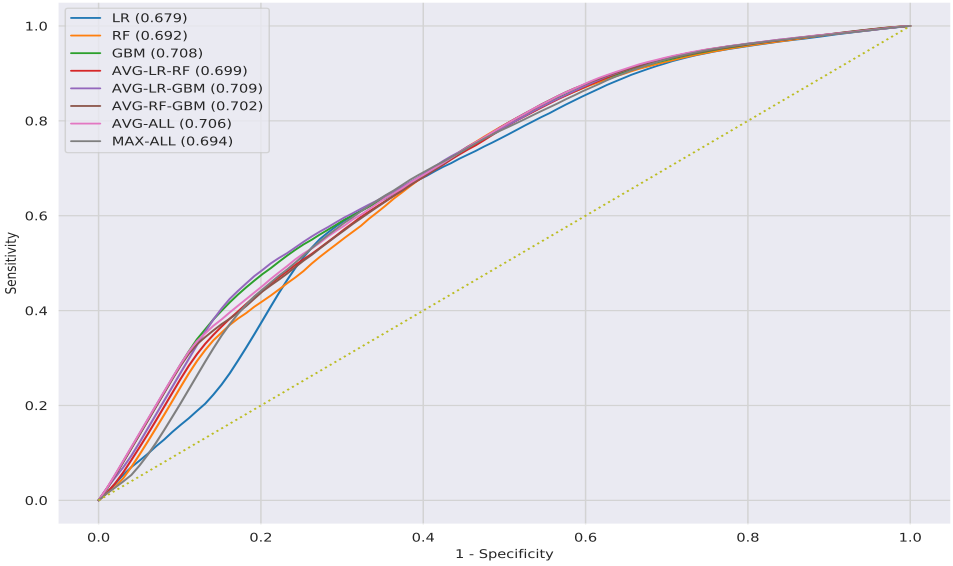


Figure B-8. Mean ROC curve for all models using structured data

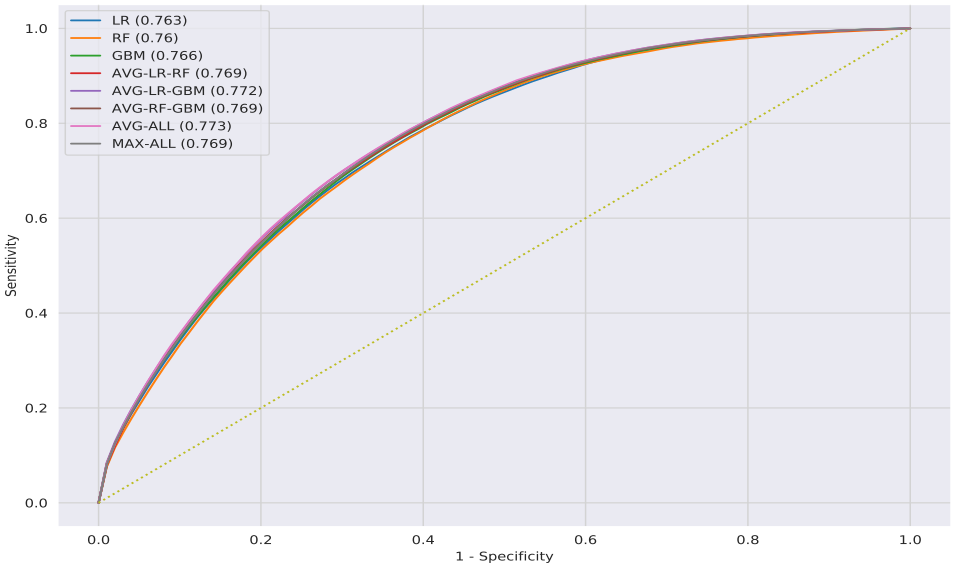


Figure B-9. Mean ROC curve for all models using unstructured data

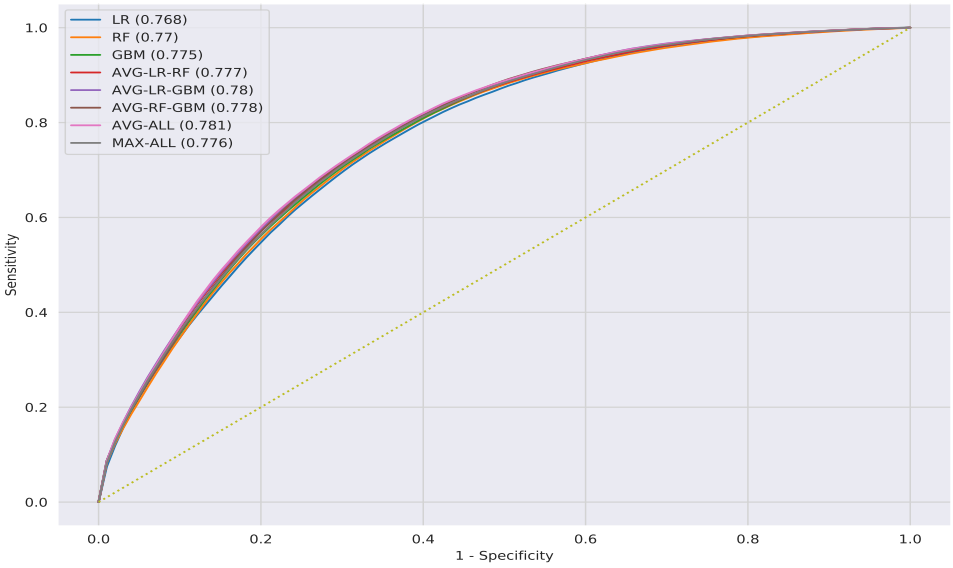


Figure B-10. Mean ROC curve for all models using multimodal data

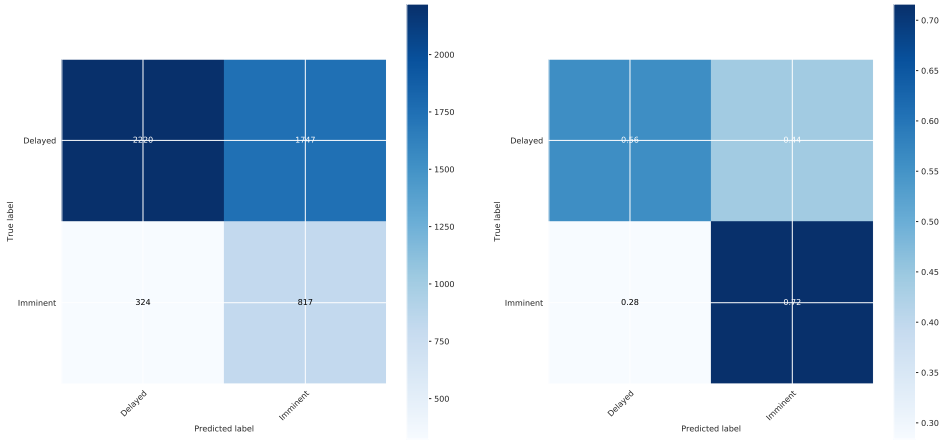


Figure B-11. Mean confusion matrix for logistic regression model using structured data

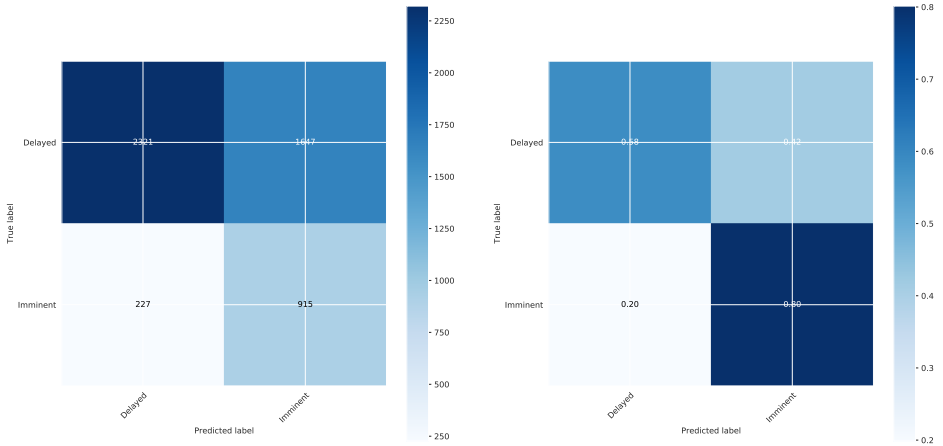


Figure B-12. Mean confusion matrix for logistic regression model using unstructured data

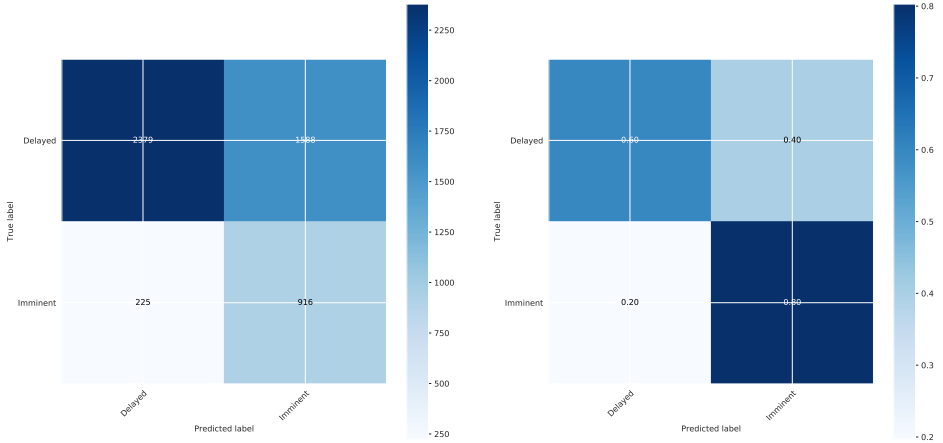


Figure B-13. Mean confusion matrix for logistic regression model using multimodal data

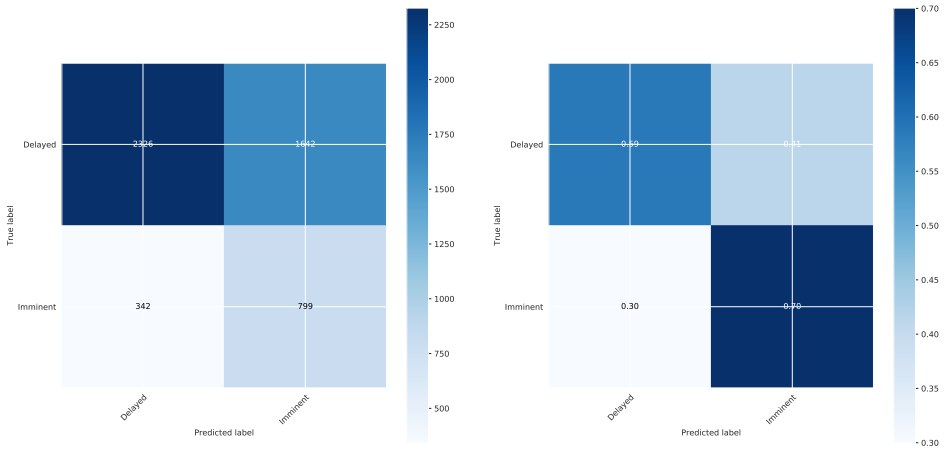


Figure B-14. Mean confusion matrix for random forests model using structured data

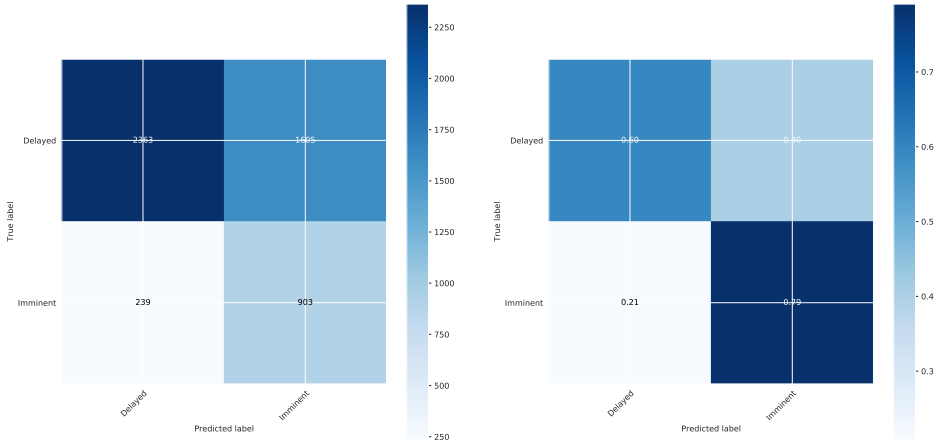


Figure B-15. Mean confusion matrix for random forests model using unstructured data

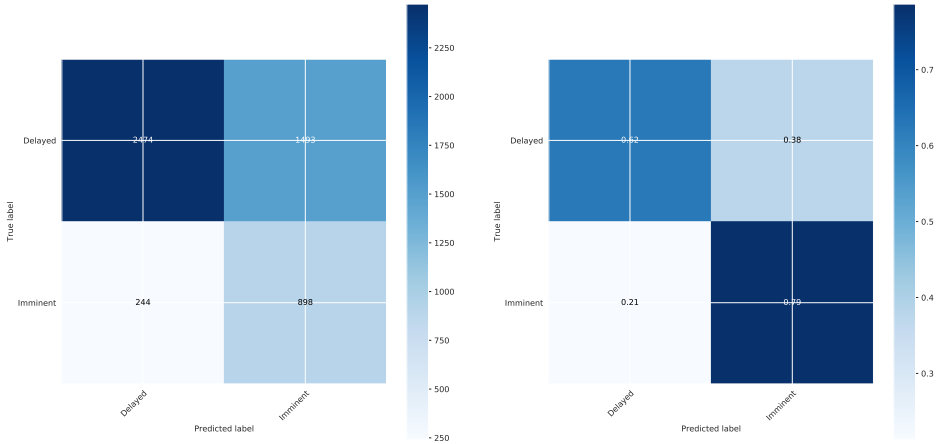


Figure B-16. Mean confusion matrix for random forests model using multimodal data

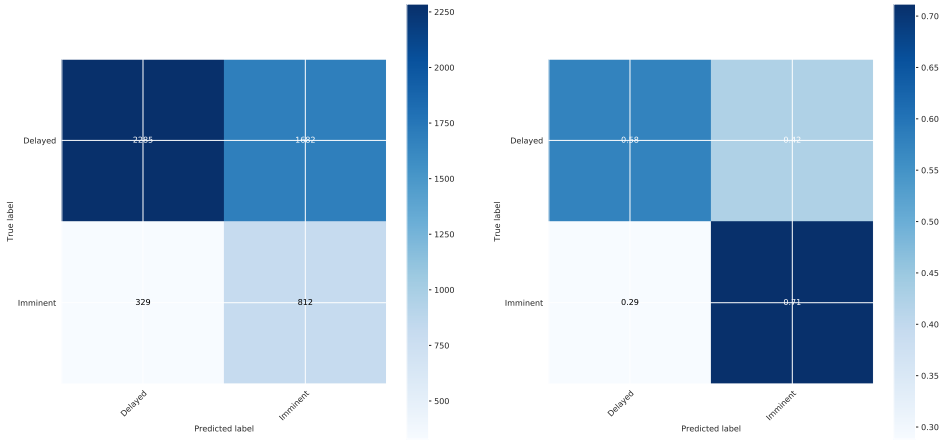


Figure B-17. Mean confusion matrix for gradient boosting machines model using structured data

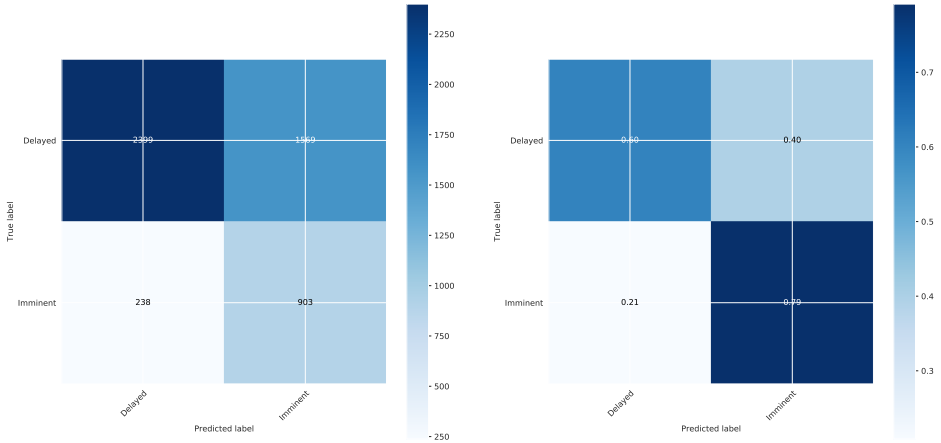


Figure B-18. Mean confusion matrix for gradient boosting machines model using unstructured data

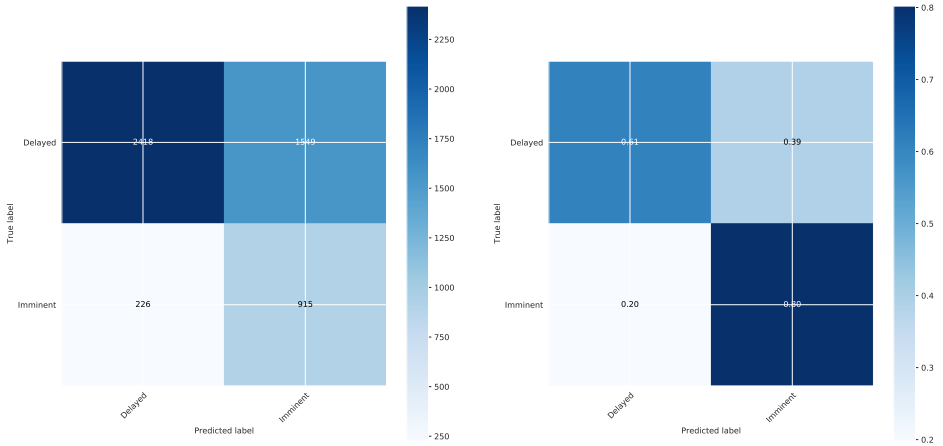


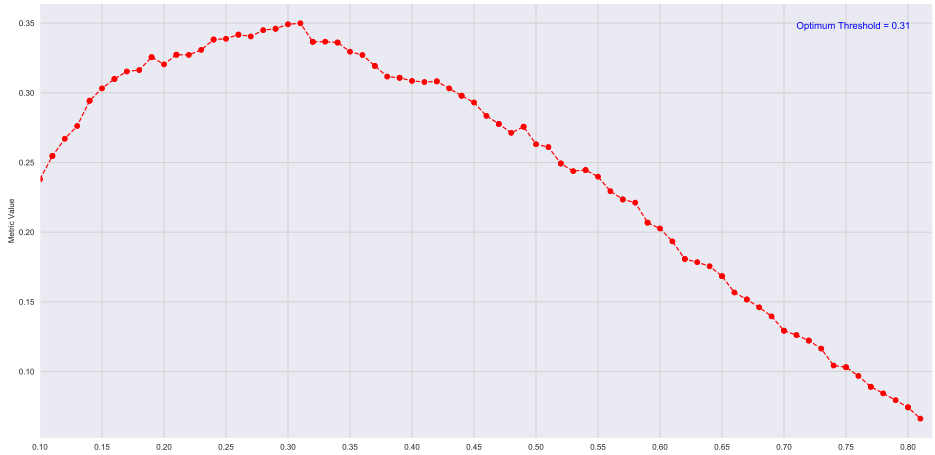
Figure B-19. Mean confusion matrix for gradient boosting machines model using multimodal data

APPENDIX C

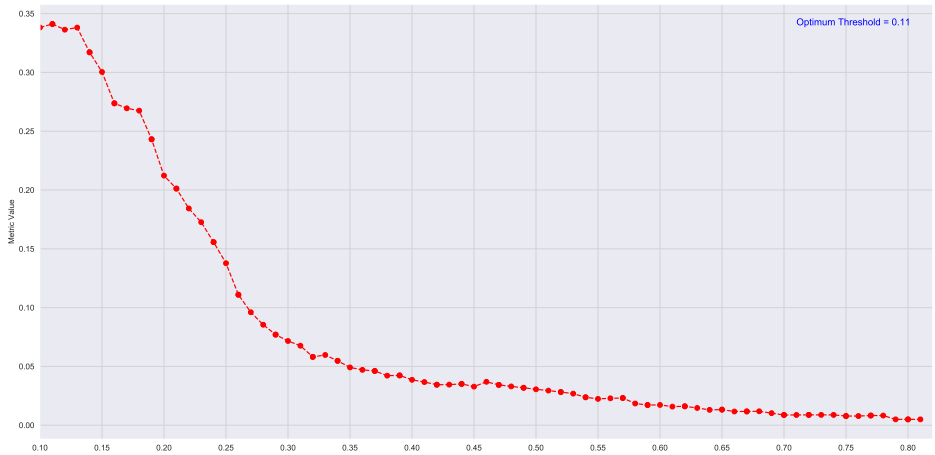
PREDICTING IMMINENT ICU ADMISSION USING MIMIC DATASET APPENDIX

Figure C-1 shows how the *Youden* index varies for the main 3 models across different discrimination thresholds. Recall that the MLH dataset is a highly imbalanced dataset with the prevalence of the positive class only 3.4%. We can see that each model has drastic variations in their *Youden* index, thus resulting in different optimum thresholds. This is reflected in figure C-2 which plots the main three performance metrics across discrimination thresholds. We can see that random forests have a very unstable PPV that shoots up towards the end with a drastic decrease in sensitivity. Random forests are built on decision trees that are sensitive to class imbalance.

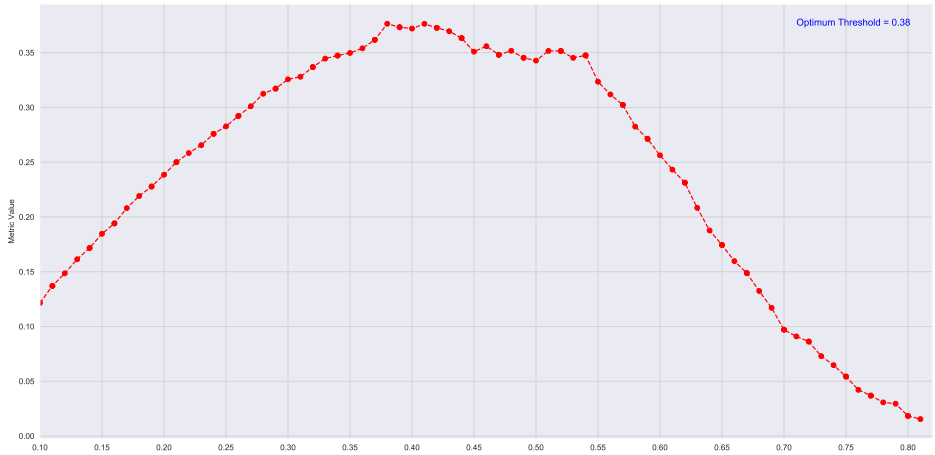
Figures C-3, C-4, and C-5 shows the average values of the confusion matrices across 100 iterations of the test set. In this, the true negative values do not mean much due to the very high prevalence of the negative value. Thus, if the model just guesses negative all the time, it will have high true negatives. GBM results in the highest percentage of true positives which correctly identifies those in need of imminent ICU admission in the next 24-48 hours. RF results in high false positives which increases worker fatigue and false negatives which results in missing positive classes.



(a)Logistic Regression

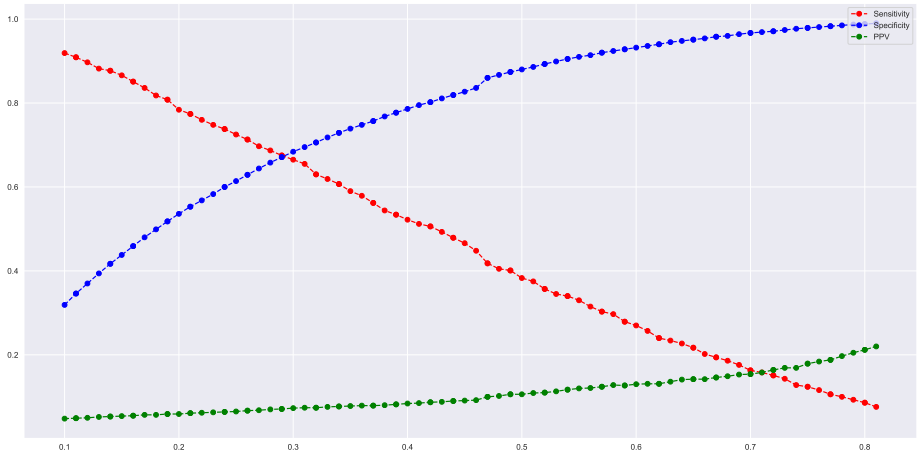


(b)Random Forests

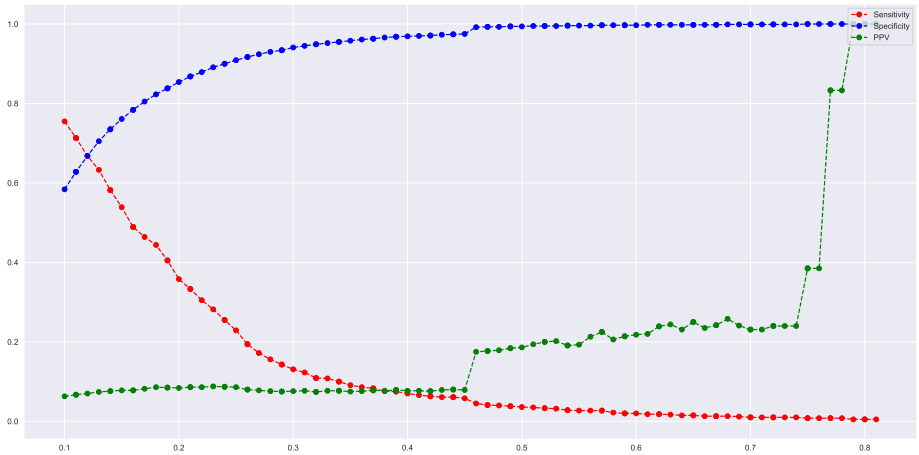


(c)Gradient Boosting Machines

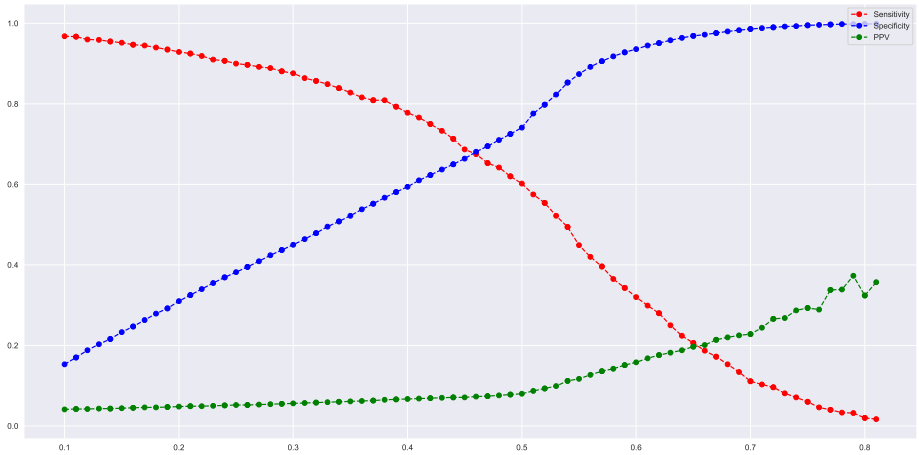
Figure C-1. *Youden* Index variation for all models across discrimination thresholds



(a)Logistic Regression



(b)Random Forests



(c)Gradient Boosting Machines

Figure C-2. Performance metrics variation
for all models across discrimination thresholds

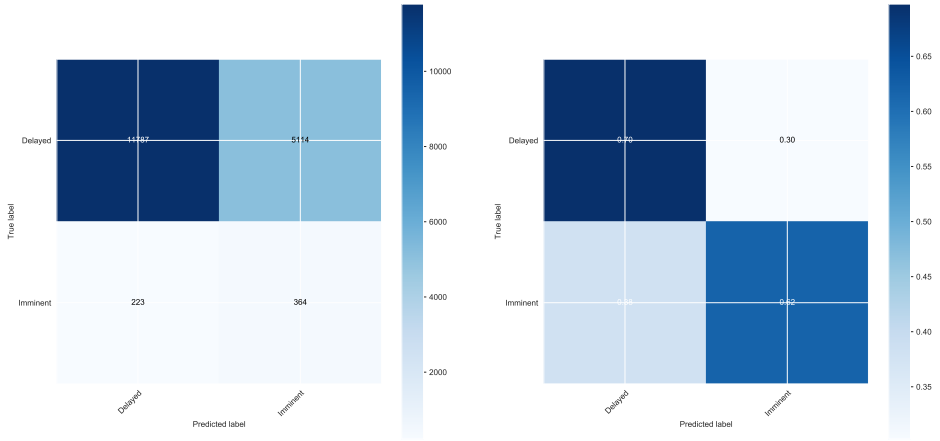


Figure C-3. Mean confusion matrix for logistic regression

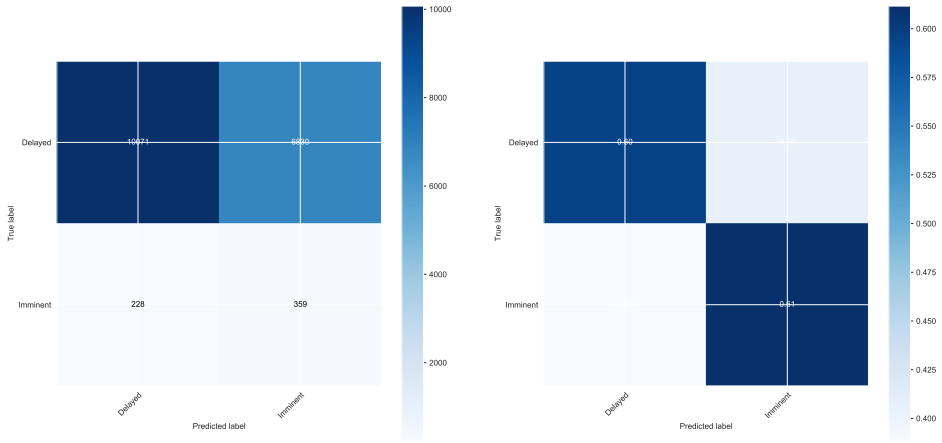


Figure C-4. Mean confusion matrix for random forests

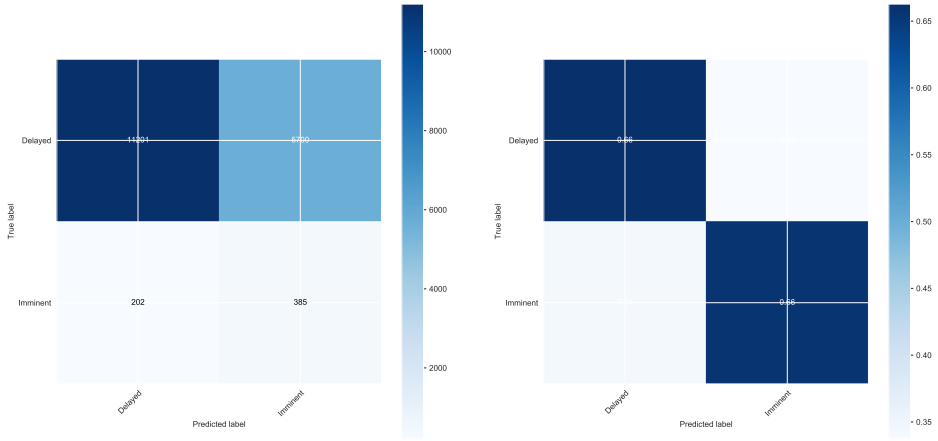


Figure C-5. Mean confusion matrix for gradient boosting machines

VITA

Sudarshan Srinivasan was born and brought up in Chennai, India. He obtained a Bachelor of Engineering in Electronics and Communications from Anna University in 2006, and a Master of Science in Computer Science from the University of Tennessee in 2011. As part of his Masters, he worked with Dr. Lynne Parker in the Distributed Intelligence Lab where he was responsible for building the software stack for various robots in the lab. He started working with Dr. Gregory Peterson in early 2017 and worked with Dr. Edmon Begoli in the ORNL research group working on the Veterans Affairs (VA) team. He was responsible for reviewing open-source software libraries that could be used for various tasks as part of the ORNL VA team.