Siamese Fine-tuning of BERT for Classification of Small and Imbalanced Datasets, Applied to Prediction of Involuntary Admissions in Mental Healthcare

Kalidas, V.

Award date:
2020

Technische Universiteit Eindhoven University of Technology

Department of Mathematics and Computer Science
Department of Industrial Engineering and Innovation Sciences
Business Intelligence Cluster
Information Sciences Research Group

Siamese Fine-tuning of BERT for Classification of Small and Imbalanced Datasets, Applied to Prediction of Involuntary Admissions in Mental Healthcare

Master Thesis

Varsha Kalidas

Supervisors:
Dr. Kalliopi Zervanou
Eline Nap, M.Sc
Prof. dr. Anna Vilanova
Prof. dr. ir Uzay Kaymak

Eindhoven, August 2020

Abstract

Electronic health records (EHRs) are recognised as a valuable source of data to support a wide range of health-care informatics use cases. Psychiatric EHRs contain information regarding the mental health status of patients in a free-text form. The goal of this thesis is to implement and evaluate suitable text processing pipelines for Dutch clinical EHRs. In particular, this thesis focuses on using NLP techniques along with Classical Machine Learning and Deep/Transfer Learning frameworks to predict involuntary admissions for patients with severe psychiatric disorders, preferably in advance. The main challenges encountered in this study pertain to working with clinical free-text, which usually involves different writing styles, varying text lengths, non-standard vocabularies, and a lack of rich annotations; and working with a very small and highly imbalanced data-set. We explore Support Vector Machines, Logistic Regression, and Decision forest classifiers from the Classical Machine Learning methodologies. Then, we explore transfer learning using the state-of-the-art pre-trained Dutch-BERT model. Furthermore, a novel fine-tuning approach based on Siamese Learning is proposed for fine-tuning the BERT architecture in order to classify small and highly skewed data-sets. Based on our experiments, we observe that performance better than random guessing can be achieved for the prediction task when using both, classical machine learning and deep transfer learning methods. We find that the classical machine learning methods outperform the deep transfer learning models as classifiers. For the deep transfer learning models, we note that in terms of model quality, the Regular and Siamese fine-tuning approach to train models as feature extractors and using classical machine learning classifiers results in better model performance than directly fine-tuning a classification model; and can be explored further for classification.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mental health disorders refer to a wide range of mental health conditions such as depression, anxiety disorders and schizophrenia, that affect a person's mood, thinking and behavior. The use of coercive measures such as involuntary admission is sometimes inevitable in managing dangerous and severely disturbing deviations of behavior that can manifest as violent, suicidal, and many other inappropriate tendencies. Although the exact definition of compulsory or involuntary admission might differ from country to country depending on the judicial context, an involuntary admission is always an admission against the will of the patient. This measure is usually employed in emergency situations where there is a high risk of danger to the patient or others and immediate treatment is necessary.

In the Netherlands, the Law for special admissions in psychiatric hospitals, Bijzondere Opnemingen Psychiatrische Ziekenhuizen (BOPZ), primarily regulates admission and allows coercive interventions to some extent under strict conditions. The law contains two different sections describing the procedures of involuntary admission. The first relates to Inbewaringstelling (IBS) or short-term involuntary admission because of immediate danger for the patient himself or others and has to be initiated by the mayor, accompanied by a written certificate of a physician. The other procedure Rechterlijke Machtiging (RM) is mandated by a judge and concerns long-term admissions aiming at treatment of patients who suffer from a severe psychiatric disorder leading to danger for others or self, including severe self-neglect or social breakdown [Steinert et al. 2014].

Coercive measures in psychiatry are a controversial topic and raise ethical, legal and clinical issues. Involuntary admission of patients is a long-lasting problem and indicates a problematic pathway to care situations within the community, largely because personal freedom is fundamentally covered by the UN declaration of human rights. Thus, there is widespread consensus that compulsory measures in psychiatry have to be considered only as measures of last resort when no other less restrictive alternatives are available [Jungfer et al. 2014].

Electronic health records (EHRs) are recognised as a valuable source of data to support a wide range of secondary informatics use cases, such as clinical decision support, observational research and business intelligence [Jensen et al. 2012]. Unlike other disciplines, free text is a key means to record information in mental healthcare as there are few laboratory tests that can describe symptoms and their severity (unlike, e.g., measuring the blood pressure for hypertension). While common conditions in mental health are represented in classification taxonomies such as the International Classification of Diseases (ICD) and Diagnostic and Statistical Manual(DSM) systems, generally speaking, it is the symptomatology of a condition that is used by clinicians to determine an appropriate treatment plan. Even when specific instruments and tests (e.g., mini mental state examination) are used, they are most often reported in free-text narrative. Mental healthcare therefore mainly relies on textual descriptions of symptoms, which are then inspected, interpreted, and assessed by health professionals in order to understand the type and the severity of the disease [Karystianis et al. 2017]. The free text portion of the EHR contains valuable clinical information which to date has not been effectively utilized for research or clinical evaluation. The key question we explore in this thesis is whether we can use machine learning

techniques to automatically process such clinical notes from the EHR in order to predict the need for involuntary admission for a given patient in advance. Such advance predictions can then be used for preemptive patient care such that the need for coercive treatments can be minimized.

1.1 Business Understanding

Antes is a non-profit mental healthcare institution based in Rotterdam, The Netherlands, that specializes in psychiatry and addiction. Since late 2017, Antes and Parnassia Groep (Bavo Europoort) have merged with the aim of jointly improving mental health care in the Rotterdam-Rijnmond region, South Holland Islands and Drechtsteden. The three core values of Antes are : Knowledgeable, Optimistic, and Respectful. The mission is to provide people with mental health problems with the right care, at the right time, and as intensively as necessary but no longer than necessary. Treatment is focused towards the recovery of adults and the elderly with serious psychiatric disorders. Antes aims at minimizing serious consequences of mental illness so that patients can regain and maintain control of their lives, and enter into social relationships according to their wishes and participate in society.

Schizophrenia and bi-polar disorder are the most common psychiatric conditions that are treated at the institution. Treatments are offered as in-patient and out-patient options and the intensity of treatment varies according to the severity of the condition. In certain patients, the use of coercive measures such as involuntary admission is sometimes inevitable in managing dangerous and severely disturbing deviations of behavior that can manifest as violent, suicidal, and many other inappropriate tendencies. Identification of such patients in advance can have clinical and policy implications for patient care, program development, and service planning.

The procedure followed at Antes for an involuntary admission is as follows: Patients with severe psychiatric disorder(s) are treated by a treating psychiatrist. In case this psychiatrist is of the opinion that a patient may need immediate involuntary admission, a second opinion is sought by requesting another examining psychiatrist to evaluate the patient. This psychiatrist is neither aware of the patient history nor has access to the patient's medical records. If the patient is evaluated to be of imminent danger to himself/herself or others, the examining psychiatrist prepares a medical certificate or 'geneeskunde verklaring' for the patient and sends it to the Mayor of the municipality of the patient. After receiving the medical certificate, the Mayor consults with the examining psychiatrist and signs the detention order if it is decided that an involuntary admission is required. The patient is then immediately admitted to the mental health facility.

In the Netherlands, there is an accelerated increase in the number of involuntary admissions seen in recent decades [Mulder et al. 2006], leading to many undesirable consequences. Many patients are not able to receive timely access to mental healthcare as they are placed on long waitlists. Additionally, the number of available beds are insufficient to cater to the patient demand. This places a burden on the government budgets to allocate more funds towards mental healthcare. Government agencies and health insurance companies may also demand for efficient and transparent use of available resources. In this context, there is a need for preventive and preemptive measures for better patient outcomes. If high risk patients can be identified in advance, appropriate interventions can be designed that reduce the number of involuntary admissions, deliver care to the patients that need the most, and ensure better utilization of resources.

In the context of a need for advanced identification of critical patients and the status of mental health-care in The Netherlands, Antes aims to determine whether it is possible to use EHR data to predict an involuntary admission before it occurs. If such an approach works, then the extra time can be utilized by the treatment staff to work out a 'crisis' plan and prevent the involuntary admission.

1.2 Problem Description

Current research in processing text data is mostly focused on the English language and existing state-of-the-art classifiers are based on using data-sets with ample amount of training examples and balanced class distributions [Friedman et al. 2013]. Given a critical problem like predicting adverse outcomes for psychiatric patients, and a small real-world clinical free-text data-set in the Dutch language with a highly skewed class distribution, we aim to investigate the possibility of developing a machine learning framework that can be used to predict involuntary admissions in psychiatric patients, preferably in advance. We investigate the use of data driven methods for this task, such that there is no need for psychiatric domain knowledge and specially hand-crafted features with the hope that the method may be applicable to broader areas of research and not just limited to this data-set in particular or only Dutch mental healthcare in general. To this end, we explore using NLP techniques to extract information from unstructured data and then using machine learning algorithms to perform the required analysis.

1.3 Research Challenges

Research in analyzing clinical free-text data is challenging because the data is usually imbalanced with respect to the class of interest [Pereira et al. 2015]. There is also a lack of publicly available data-sets and current research is conducted on private institutional data [Wang et al. 2018]. Such data tends to be smaller when compared to data used in general NLP research, and thus presents another challenge when working in the health-care domain.

While unstructured clinical texts store a lot of valuable patient information, they are very subjective to the doctor or the nurse writing them and lack common structural frameworks. There could also be many errors related to language use such as improper grammatical use, short phrases, local dialects, and semantic ambiguities, which increase the complexity of data processing and analysis. A related challenge includes the extensive use of negations to rule out clinical signs and references to subjects other than the actual patient.

For our study, some critical challenges that may possibly undermine the performance of machine learning(ML) algorithms are the use of abbreviations, the particular writing style of the clinician, whether multiple persons make observations on the same patient, and patient behaviours/outcomes related to the person making the observation itself.

While commonly recognized abbreviations from the medical literature can be incorporated and accounted for by an ML algorithm, use of personal and custom abbreviations that are specific to a clinician becomes difficult to analyse without input from the person that made the observation. This issue becomes complicated if multiple persons use multiple personal abbreviations to make observations on the same patient.

Since there is no established structural framework or a specific vocabulary for reporting patient observations, the writing style used by the clinicians varies considerably. This variation in style itself may introduce variations in the degree of severity of the symptom that is recorded in the EHR. Additionally, the severity of a symptom is based on the subjective perception of the clinician. For instance, one clinician may record a facial expression as 'sad' while another may record it as 'mildly depressed'. It is possible that certain clinicians tend to either over-estimate the severity and recommend an involuntary admission or alternatively, under-estimate the severity and fail to identify a patient that needed an involuntary admission. The problem gets compounded when different clinicians with different writing styles make notes on a single patient. Thus, it becomes difficult for a ML algorithm to learn to predict correctly.

In rare cases, it is also possible that a patient reacts differently in the presence of different clinicians that results in different mental health evaluations for a patient. This again introduces a modelling challenge since the data and outcome would differ based on the clinician.

In these above mentioned cases, the clinicians may become confounding factors. They may affect the data and the outcome and introduce spurious associations.The fact that multiple confounding clinicians may make observations only makes it more confusing for a ML model to learn

and predict correctly.

Furthermore, it should be noted that in practice, a patient is evaluated for involuntary admission by a psychiatrist who has no information about the patient's history. The patient's reference to involuntary admission is not made based on the EHR text but on the perception of imminent danger by a non-treating psychiatrist.

The rest of this thesis is organised as follows: Chapter 2 provides a literature review of related work in the domain of mental healthcare and an exploratory analysis of suitable NLP tasks, classical machine learning, and deep learning methodologies that may be explored in this thesis. Chapter 3 gives a description of the data-set provided by Antes Groep that we use for the prediction task. In chapter 4 we describe the methodological frameworks that we explored for our research. Section 4.2 provides a description of the various NLP tasks that were used for pre-processing the raw text data, and making it suitable for use by machine learning algorithms. Then, in section 4.3 we present various classical machine learning and deep learning frameworks that we considered suitable for the prediction task. We present our proposed Siamese Fine-tuning approach in this chapter. Then we present our experimental setup and results in chapter 5. We conclude the thesis by reflecting over the contributions and limitations of this work, and possible future research directions in chapter 6.

Chapter 2

Literature Review

Using natural language processing (NLP) techniques along with machine learning algorithms to analyze data from EHRs is emerging as a promising approach for applications such as the identification of health conditions [Melton and Hripcsak 2005] and prediction of adverse outcomes such as post-operative complications [Murff et al. 2011].

2.1 Research on using EHR data in Mental Healthcare

In the mental health domain, there have been studies for predicting psychiatric admissions [Friedman et al. 1983; Lyons et al. 1997; Olfson et al. 2011] but these make use of structured variables like medical codes, patient demographics, medication history, and other risk factors. Research on suicide risks indicates that the predictive value of combinations of risk factors obtained from structured EHR fields becomes asymptotic due to the risk conferred by multiple risk factors being less than the sum of each individual risk factor [Conner et al. 2012]. Therefore, extracting information from the unstructured clinical texts present in the EHR may help to build more useful prediction models in the mental health domain.

Free-text along with patient demographics and structured variables from the EHR has been used in performing diagnosis of suicide [Cook et al. 2016] and depression [Huang et al. 2014]. However, these models were used to differentiate patients that were likely to have mental issues from patients that were mentally healthy. In such a scenario, structured variables such as medication history, patient health questionnaires, and demographics are likely to have sufficient discriminatory power for the particular analysis. For predicting involuntary admissions in patients with schizophrenia and bi-polar disorders, structured variables may not have enough discriminatory strength as all patients are already diagnosed with the disorder and are likely to have similar values for the structured variables. Additionally, the use of demographic data in such a scenario may increase the risk of introducing unintended bias into the model.

Clinical observations and notes made by nurses, doctors and other health-care professionals in a free-text format are likely to contain valuable information that may help with predicting admissions since deviations from some normal behaviour precipitate the need for involuntary admissions. Such deviations are highly likely to be recorded in the patient's case record. Research using unstructured text from the EHR for mental health-care is still in its nascent stages and there have been very few studies in this area. Suicide risk among veterans was predicted using only the free-text from the EHR by Poulin et al. [2014]. More recently, Menger et al. [2018b] used clinical text for predicting inpatient violence incidents in treatment facilities. In this study, we contribute to this very nascent research area by investigating the use of EHR text to predict involuntary psychiatric admissions.

Clinical NLP systems are built on the foundation of words or phrases as medical terms to represent the domain concepts and understanding the relations between these identified concepts [Demner-Fushman et al. 2009]. Traditional NLP tools follow a knowledge-driven methodology

based on semantic and lexical rules combined with manually constructed dictionaries [Eriksson et al. 2013]. It has often been necessary for a new NLP tool to be developed or adapted for each medical database, and even for each clinical question, when processing EHR free-text [Ford et al. 2016]. This is labor intensive, as it requires significant clinical knowledge and also requires that the tools be tested on significant amounts of text annotated by human experts. In the present study, since there is a lack of specialist domain knowledge and specially annotated text, we use NLP techniques to pre-process the data into a form that can then be used by more general classification algorithms. NLP methods like tokenization, stop word removal, and part-of-speech tagging can be used to pre-process the text to identify terms or relations depending on the context and analytical goals [Manning et al. 2008]. Next, the text can be represented in terms of appropriate features. Then suitable data mining techniques are used to perform the analytical task.

The most successfully used methods for text mining in general and mining EHR data in particular are support vector machines(SVM), logistic regression, naive Bayes, and decision trees [Abbe et al. 2015; Aggarwal and Zhai 2012]. For this study we start by exploring some of these classical machine learning methods in order to examine their suitability for use in our clinical data-set, as well as to establish a reference for baseline model performance against which we could compare other proposed solutions.

2.2 Deep Learning and EHR Data

While the use of classical machine learning methods has been widely established in classification tasks, novel Deep Learning [Hao et al. 2016] techniques have also been utilized recently for text classification. In exploring EHR data, various Deep Learning techniques act as powerful feature discriminators and show promising results for applications such as information extraction, outcome prediction and de-identification [Shickel et al. 2017]. When dealing with text data, the classical methods do not generalize well to new texts which may include words that were not used to train the model. On the other hand, deep learning techniques like Convolutional neural networks(CNN) [LeCun et al. 1995] and Recurrent Neural Networks(RNN) [Elman 1990] that make use of dense word embeddings to represent text are able to generalize well to words not seen in the training vocabulary [Goldberg 2016]. An additional advantage of using deep learning techniques is that they can analyze text as a sequence of words, allowing for a richer representation of the input. In contrast, the popular Bag-of-Words(BOW) model [Manning et al. 2008] used to represent text for classical machine learning algorithms completely disregards the order of words.

For text classification, RNNs process a text sequentially, learn an internal fixed sized encoding of the input, and determine class based on this encoded representation. Similarly, CNNs also learn a fixed encoding of the input but do so by using the convolution operator on the input sequence, applied in a sliding window manner. While CNNs are an improvement over the BOW model, they are sensitive to mostly local patterns and not to the order of patterns that are far apart in a sequence. RNNs are capable of paying attention to structured patterns across the entire sequence. Since the present study uses clinical notes for prediction, it is expected that information may be spread out over the entire text and hence RNNs may be more suitable. For the present study, since we have a very small data-set, training CNN and RNN models from scratch becomes unfeasible, and we turn to using deep transfer learning approaches instead. Next, we briefly describe the problem of imbalanced data-set classification before moving on to transfer learning.

2.3 Classification of Imbalanced Text Data

Imbalanced classification is a challenge in text classification and although many strategies have been proposed in the non-text domain, they were not effective when applied to text data [Sun et al. 2009]. The commonly used strategies are: (i) resampling: under-sampling negative examples or over-sampling positive examples to re-balance the training examples; (ii) instance weighting: assigning different error-classification costs to negative and positive training examples during clas-

sifier training; and (iii) thresholding: adjusting decision thresholds of a classifier to balance the precision and recall. Experiments by Sun et al. [2009] showed that the best decision surface was often learned by the standard SVM, without any of the proposed strategies. Hence, we investigate the use of the standard classical machine learning classifiers for this study, and explore transfer learning for further improvements in classification performance.

2.4 Transfer Learning

The problem of classifying small data-sets can be approached using transfer learning [Aggarwal and Zhai 2012]. In this framework, knowledge is extracted from some auxiliary domain to help improve the learning in a target domain. In the context of using classical machine learning algorithms for text classification, we find that such cross domain transfer learning starts with the underlying assumption and requirement that the data are represented with the same feature space for both auxiliary and target learning domains. Since our study uses psychiatric clinical text in Dutch, it is extremely difficult to find a suitable auxiliary data-set for cross domain transfer learning. Hence, we turn towards deep transfer learning in order to find a solution that does not need such suitable data-sets.

2.5 Deep Transfer Learning in NLP

Transfer learning in deep learning networks is different from that in classical machine learning methods and may be considered for use in text classification. Features in deep neural networks in computer vision(CV) have been observed to transition from general to task-specific from the first to the last layer [Yosinski et al. 2014]. Therefore, most transfer learning methods in CV focus on transferring the first layers of the model [Long et al. 2015]. Razavian et al. [2014] achieved state-of-the-art results using features of an ImageNet model as input to a simple classifier. This approach was superseded by fine-tuning either the last [Donahue et al. 2014] or several of the last layers of a pre-trained model and leaving the remaining layers frozen [Long et al. 2015].

Transfer learning research in deep learning related to NLP is largely focused on fine-tuning language models with pre-trained word embeddings [Mikolov et al. 2013]. While this approach has had a large impact and is used in most state-of-the-art models, word-embeddings are used only in the first layer and the entire model needs to be trained from scratch using a large number of examples. In addition, Mou et al. [2016] find that the transferability of a neural network model in NLP largely depends on the semantic similarity of the tasks and leads to catastrophic forgetting in case of dissimilar tasks. They also find that the output layer is mainly specific to the data-set and is not transferable. For our study, since pre-trained clinical psychiatric word embeddings in the Dutch language are not publicly available, we search for other transfer learning techniques instead.

Howard and Ruder [2018] introduced Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method that can be applied to any task in NLP, and introduce techniques that are key for fine-tuning a language model for a new task such that the model does not forget what it previously learnt. The language model is first trained on a large amount of general data and then fine-tuned for another task. Training a language model requires a lot of computational power which is out of scope of a master project and hence this type of transferability cannot be considered due to feasability reasons. In addition, the publicly available ULMFiT model is trained for English and hence is unsuitable for a Dutch data-set. Eisenschols et al (2019) introduced Efficient Multi-lingual Language Model Fine-tuning (MultiFit) for language modelling beyond English [Eisenschlos et al. 2019]. However, a model for Dutch language is still not publicly available.

More recently, pre-trained language models such as BERT (Bidirectional Encoder Representations from Transformers) have shown to be useful in learning common language representations by utilizing a large amount of unlabeled data [Devlin et al. 2019]. BERT uses transformers based

on bidirectional conditioning to learn contextualized word embeddings [Peters et al. 2018]. When compared to embeddings such as word2vec and GloVe that capture only semantic and syntactic relationships, BERT computes contextualized embeddings that are able to also take into account the context of a word. Thus, BERT language models are able to handle polysemy as well.

Sun et al. [2019] demonstrated that BERT models could be fine-tuned to achieve state-of-the-art results in classification tasks on English data-sets. They also demonstrated its usability for as few as 200 training examples. However, these tasks were performed on balanced data-sets. Recently, multilingual BERT models have been made public and hence a BERT language model exists for the Dutch language. For our study, we would first investigate whether a general language model like BERT(Dutch) could be used as classifier for a clinical data-set in Dutch. Further, inspired by one-shot learning for image classification using the Siamese architecture where a model learns discriminative features using very few training examples [Koch 2015], we propose to investigate whether such an architecture could be used for classifying our small and imbalanced text data-set. To handle the problem of class imbalance, we propose to fine-tune BERT(Dutch) using Siamese learning and then use it as a document feature extractor. The fine-tuned model can be used to represent documents in the form of embeddings and these document representations can then be used as input features for use in a simple classical machine learning classifier like an SVM.

2.6 Conclusion

We find that there is no other study that investigates using free-text from the EHR to predict involuntary admissions in patients with severe psychiatric illnesses. We would be the first to investigate this prediction objective in general, and for Dutch clinical texts in particular. Due to the lack of a richly annotated data-set incorporating expert psychiatric knowledge, it is decided to limit the use of NLP techniques to the data pre-processing stage and to use more general classification algorithms for the prediction task. Next, we find that the challenge of working with a small and highly imbalanced data-set could also be approached using the state-of-the-art deep transfer learning approaches. In this regard, pre-trained BERT language models could be used as classifiers and feature extractors to investigate their predictive potential. Classical machine learning algorithms can be used as a baseline to evaluate the performance of the deep transfer learning models.

Chapter 3

Data Description

In this chapter we provide a detailed description of the data-set that we used in order to train a model that can predict involuntary admissions. We also provide an overview of the privacy, legal and ethical considerations that were taken into account.

The data for this study was provided by Antes Groep. Patients diagnosed with schizophrenia or bipolar disorder were selected for this study. These patients were in contact with Antes for at least 30 days in 2016. They were followed through the years 2017, 2018, and 2019 to check whether an involuntary admission took place. The patients were then divided into two groups: patients with and without involuntary admission. The health records starting 2016 could be extracted from the EHR system for analysis. Many patients with involuntary admissions in 2016 had historical records but these records were in another format and could not be extracted for analysis. Therefore, patients with involuntary admissions in 2016 were excluded from the analysis so that sufficient historical data was available. Nine advance prediction time-frames corresponding to 0, 3, 7, 10, 14, 30, 60, 90 and 180 days before involuntary admission were selected for the analysis. These time-frames were selected by Antes as being relevant for the predictions. The extraction, de-identification, and organization of the data was done by Antes.

The data-set was a collection of health records for the selected patients and was organized into two fields. One field contained the EHR text, and the other contained information about the involuntary admission in a binary form, where a value of 1 corresponded to an involuntary admission and a value of 0 corresponded to no involuntary admission. The EHR text was a concatenation of all reports from a patient's EHR. The reports were in the form of textual observations entered by the care givers (psychiatrists, psychologists, nurses etc), both at the moment of intake, and during treatment. They were all written using the Dutch language.

The data was hosted in the Data Science Environment of the institution. In accordance with the data confidentiality policy of the company, remote access was provided to analyze the data. A total of 9 data-sets corresponding to the nine time-frames were made available in the pickle [1] format. The text for the different time-frames was selected as follows :

- 0 days before admission: concatenation of all text reports starting from 2016 till the start of the involuntary admission
- 3 days before admission: concatenation of all text reports starting from 2016 till 3 days before the start of the involuntary admission
- 7 days before admission: concatenation of all text reports starting from 2016 till 7 days before the start of the involuntary admission
- 10 days before admission: concatenation of all text reports starting from 2016 till 10 days before the start of the involuntary admission

[1] Python module implementing binary protocols for serializing and de-serializing a Python object structure. https://docs.python.org/3/library/pickle.html

Data-set (days before admission)	Number of Positive examples	Number of Negative examples
0 days	187	2249
3 days	185	2249
7 days	185	2249
10 days	185	2249
14 days	185	2249
30 days	182	2249
60 days	182	2249
90 days	180	2249
180 days	175	2249

Table 3.1: Class distribution for all data-sets, before data pre-processing

- 14 days before admission: concatenation of all text reports starting from 2016 till 14 days before the start of the involuntary admission
- 30 days before admission: concatenation of all text reports starting from 2016 till 30 days before the start of the involuntary admission
- 60 days before admission: concatenation of all text reports starting from 2016 till 60 days before the start of the involuntary admission
- 90 days before admission: concatenation of all text reports starting from 2016 till 90 days before the start of the involuntary admission
- 180 days before admission: concatenation of all text reports starting from 2016 till 180 days before the start of the involuntary admission

Descriptive statistics were computed for each of the data-sets. Table 3.1 shows the distribution of the examples across the two classes. We note that this is a very imbalanced data-set with the positive examples representing only 7% of the data-set on average. Next, Table 3.2 shows the token-level text statistics for the different data-sets. We note that while there are some very long texts, there are also texts with just a single token. These texts will be scrutinized at the data preparation to see whether they provide any meaningful information. We note that the type-token ratio for the data-sets is 0.0082 on average. The type-token ratio(TTR) is a measure of lexical richness, or variety in vocabulary. The closer the TTR ratio is to 1, the greater the lexical richness of the segment. It varies very widely in accordance with the length of the text and generally decreases with increasing text lengths since the longer a text runs on, the fewer novel words will be introduced. Next, we note that the mean sentence length is very high. This could be because the text still contains different special characters that would all be counted as tokens.

Privacy considerations

The use of patient data for research puts a strain on patient privacy, since this requires using the data out of the context of health care. This entails, for example, copying the data to different databases, where it can be accessed by data analysts. Medical staff is allowed to see patient information under medical confidentiality but technical staff such as data managers or data analysts typically do not have a treatment relation with the patient, and therefore should not be able to identify individual patients in a research data-set.

From a patient perspective, protecting the private details of a disease from the public is essential in retaining the trust bond between a physician and the patient. Any violation of this confidentiality can therefore have serious consequences for the relation between a healthcare institution and a patient. A patient may be averse to their data being used for research and might

Data-set (days before admission)	Corpus Size (Total Tokens)	Number of Unique Tokens	Type-Token Ratio	Mean Sentence length (in tokens)	Minimum record length (in tokens)	Maximum record length (in tokens)	Mean record length (in tokens)
0 days	47215351	386120	0.008178	1652	1	266835	19382
3 days	47090026	385445	0.008185	1650	1	266835	19346
7 days	46999008	384870	0.008188	1647	1	266835	19309
10 days	46954096	384589	0.00819	1645	1	266835	19290
14 days	46897041	384244	0.00819	1644	1	266835	19267
30 days	46720212	383378	0.0082	1640	1	266835	19218
60 days	46493037	382212	0.00822	1632	1	266835	19125
90 days	46311103	381224	0.00823	1627	1	266835	19065
180 days	45885013	378838	0.008256	1617	1	266835	18929

Table 3.2: Text Statistics, before pre-processing

even consider seeking treatment elsewhere. Moreover, a potential data breach may expose private patient information to the general public. Therefore, appropriate privacy considerations must be made before the start of the research.

The patient data used in this study was de-identified at Antes by the DEDUCE algorithm [Menger et al. 2018a] for legal and privacy considerations. The following Personal Health Information (PHI) data were removed:

- Person names, including initials
- Geographical locations smaller than a country
- Names of institutions that are related to patient treatment
- Dates
- Ages
- Patient Numbers
- Telephone numbers
- E-mail addresses and URLs

Legal Considerations

From 25 May 2018, the General Data Protection Regulation (GDPR) has been directly applicable in all Member States of the European Union. In addition to these regulations, EU Member States may maintain or introduce further conditions, including limitations, with regard to the processing of genetic data, biometric data or data concerning health. According to the GDPR, health data[2] refers to personal information (also called personal data) that relates to the health status of a person. This includes both medical data (doctor referrals and prescriptions, medical examination reports, laboratory tests, radiographs, etc.), but also administrative and financial information about health (the scheduling of medical appointments, invoices for healthcare services and medical certificates for sick leave management, etc.). Health data is considered sensitive data and is subject to particularly strict rules and can only be processed by health professionals who are bound by the obligation of medical secrecy. Furthermore, the organisation is required to take the necessary security measures to ensure that the health data is protected and not subject to any unauthorised disclosure. In compliance with the GDPR, a data confidentiality contract was

[2]https://edps.europa.eu/data-protection/data-protection/reference-library/health-data-workplace_en

agreed to by the research student. In addition, the student provided a report of the data access made, along with the access purpose, on a daily basis to the company supervisor.

In the Netherlands no specific laws on the reuse of medical data exist, but there are general rules for dealing with personal data, that can be applied to medical data as well. Since EHR data is used for retrospective research, is not specifically collected for research purposes, and human subjects are only indirectly involved, the Medical Research Involving Human Subjects Act (WMO) does not need to be taken into account. Only the Agreement on Medical Treatment Act (WBGO) and the GDPR play a role in this situation. Retrospective research with medical records needs to be proposed to the Medical Ethics Committee (METC), which verifies that the proposed research is in line with privacy legislation. An exception to this is when only anonymized data is used, which is the case if the de-identification process is executed perfectly.

For the purpose of this study, since the data was de-identified, approval of the METC was not required. The Privacy Officer at Antes had to approve the research.

Ethical Considerations

A number of ethical dilemmas arise when working with patient data. According to Wade [2007], collecting and using patient data, beyond making an individual clinical decision, may be ethically justified only if: there is (or could reasonably arise) a question to be answered; the methodology (design, data collected, etc) will answer the question; and the costs, including both communal healthcare resources and any risks and burden imposed on the participants, justify the benefits to society. In contemplating these decision factors, another difficult dilemma arises: who should ask the questions, and who should make the ethical judgment?

While such philosophical dilemmas regarding the decision factors and decision makers are challenging to address, some ethical concerns of a practical nature were addressed sufficiently. The two primary ethical concerns pertaining to research based on medical records are obtaining informed consent from patients and maintaining the confidentiality of data subjects [Ashwinkumar and Anandakumar 2010]. Under the Netherlands' Agreement on Medical Treatment Act (Dutch abbreviation WGBO), patient record research does not require patients' informed consent if individual patients cannot be identified on the basis of the data. In this study, since the patient data is de-identified, no informed consent was required. Further, as per the applicable Dutch laws, all researchers involved in the analysis were bound to strict confidentiality, thereby maintaining the data subject confidentiality.

In addition to ethical concerns around data use, certain undesirable consequences of creating models to predict involuntary admissions must also be analysed. With the availability of advance prediction models, care providers might resort to direct action rather than increasing therapeutic attention towards the patient. Hence, proper measures should be in place to ensure that such models are used only to prevent adverse patient outcomes and not for preemptive coercive treatments.

Chapter 4

Experimental Methodology

4.1 Overview

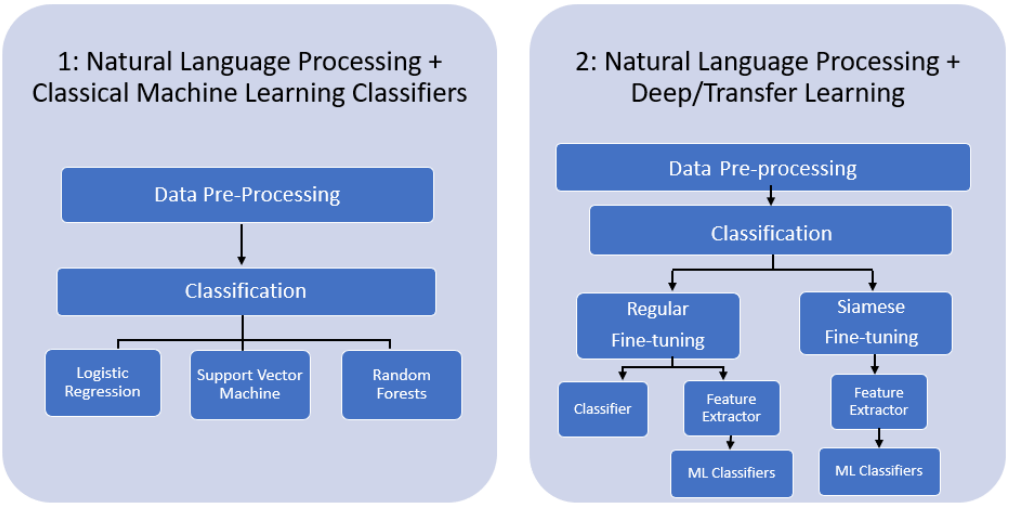

Figure 4.1: An overview of the experimental methodologies

In this chapter, we provide a high level view of the experimental methodologies that will be used in this project. There are two broad sections of methods that are explored here. Figure 4.1 provides an organization of these methods. The first section relates to using NLP along with classical machine learning classification algorithms to perform the classification task. We explore Logistic regression, Support Vector Machines, and Random forest classifiers here. The second section relates to using NLP along with deep transfer learning in order to classify the data. In this section, we explore two transfer learning techniques for training the classifiers. One is the regular fine-tuning technique where a pre-trained model is fine-tuned on the current data-set as a classifier. Here, we fine-tune a pre-trained model to use in two configurations: as a direct classifier and as a feature extractor. The other technique that we propose and explore is fine-tuning a pre-trained model in a Siamese manner as a feature extractor. When using models as feature extractors, the pre-trained model is fine-tuned such that it can be used to represent the input text as features and classical machine learning algorithms can then use these representations to perform the classification. For both sections the given data-set is first pre-processed before training a classifier. These pre-processing techniques based on NLP are described in section 4.2. The pre-

processing for classical ML classifiers is described in 4.2.1 and that for deep transfer learning based classifiers is described in 4.2.2. The various modelling methodologies are then described in section 4.3.

4.2 Data Preparation

Text documents in natural language are in an unstructured form as strings of characters. This form of data cannot be directly analysed by classification algorithms. These algorithms require the data to be in a numerical form. Hence, the text data needs to be pre-processed into linguistically significant and methodologically useful units or features that can be interpreted and analysed by the classification algorithms. In order to achieve this, various tasks can be performed, and they largely depend on the data analysis goals. For the purpose of predicting involuntary admissions from EHR data, a pre-processing pipeline consisting of various NLP tasks will be implemented for cleaning the data and selecting text features. The goal is to eliminate noise and select a compact meaningful representation of the input for efficient classification. In the rest of this chapter, we provide a motivation and explanation of the various NLP tasks that were applied to the data-set described in chapter 3.

4.2.1 Data Preparation for Machine Learning Models

In this section we describe the pre-processing pipeline that we used for preparing the text data to be analysed by the classical ML algorithms. This pipeline is illustrated in figure 4.2 and each step is explained in the following sub-sections.

Removal of Very Short Text Records

In the given data-sets, there may be texts that are too short for any meaningful analysis. In our dataset, some records had a length of just one token (see table 3.2). On examining such short records, it was found that some records just had one number in the text field, and others consisted of a single sentence 'Refer to patient dossier'. Since such texts have no information to help with the prediction task, these records must be removed.

Removal of Numbers, Special Characters, Single and Double Characters

After selecting appropriate records for analysis, we noted that there were words with tags like <Persoon>. We also noted that some patients had some inclusion of diagnostic results in their records and many special characters like +, -, (, and) were used. Since diagnostic abbreviations, numbers, and special characters are not expected to contribute to the present study, all special characters, numbers and strings of single or double characters must be removed.

Tokenization

Tokenization is the task of splitting up a given character sequence from a defined document unit into sub-units, called tokens [Manning et al. 2008]. A token is a sequence of characters that are grouped together as a useful semantic unit for a particular task. The splitting is done after locating the token boundary. A token boundary defines the end of one token and the beginning of the next token. Token boundaries are dependent on the language, task and type of document, and are usually white spaces or punctuation characters. The resulting list of tokens is used as input for further processing. A token is different from a word in the sense that a word is a string of alphabetical characters while a token can be a string of any type of characters (alphabets, numbers, special symbols, etc). For example, consider the following sentence : Welcome to Eindhoven University !. After tokenization using the white space as the token boundary, the resulting list of tokens or vocabulary is : 'Welcome', 'to', 'Eindhoven', 'University', '!'. The exclamation character is a token but is not a word.

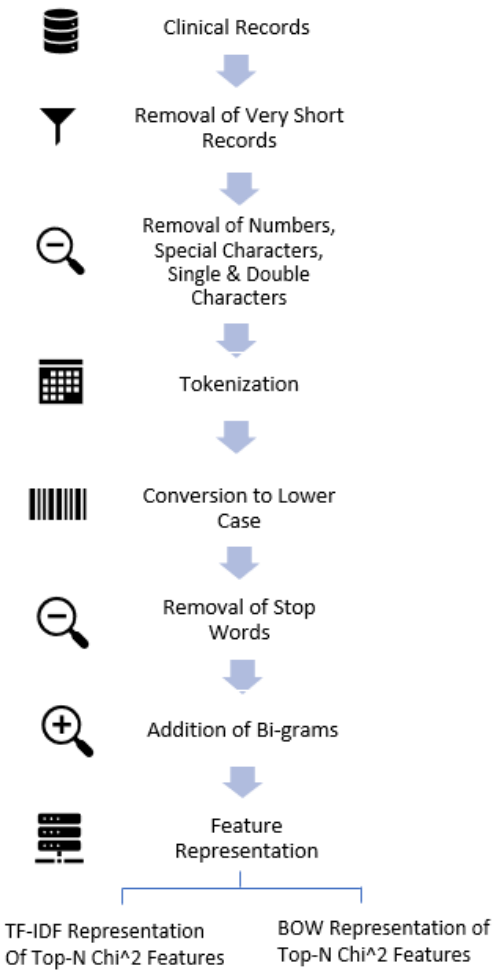

Figure 4.2: An overview of the data pre-processing pipeline for classical machine learning algorithms

Case Folding

In this task, all letters are reduced to lower-case. This is done in order to normalize the tokens into the same case form so that superficial differences between tokens can be ignored. This helps with better analysis since available information is properly utilized. For instance, suppose a document contains the words 'good', 'Good', and 'GOOD'. Without case-folding the three words are treated as different words, which reduces the quality of information that can be extracted.

Removal of Stop Words

Text documents usually have a number of common words like 'a', 'an' and 'the' that are very frequent and uniformly distributed across all documents in a corpus. Such words have little semantic value and do not add value in discriminating one document from the other. Hence they are removed from the vocabulary. A list of stop words is called a stop list. Such a list is usually determined by first sorting the terms by the total number of times each term appears in the document collection, and then selecting the most frequent terms, often filtered manually for

their semantic content relative to the domain of the documents being indexed, as a stop list. The publicly available list of stopwords for the Dutch language was used. Furthermore, it is possible that clinicians use negations to note down certain symptoms such as 'not talking'. Therefore certain words from the stop list must still be included to not lose meaningful information. A list of all the Dutch stopwords is given in Appendix A at A.

Addition of Bi-grams

A bi-gram is a two-word sequence of words [Manning et al. 2008]. When using classical machine learning algorithms like SVMs for classifying text data, word/token order is not taken into account by the classifier. Tokens are independent of one another and this may reduce the quality of analysis. For instance, in the context of mental healthcare, the care provider may note the absence of certain behaviours in the patients. Remarks like 'not depressed' or 'not friendly' are highly likely to be present in the patient record. Ignoring such sequences may lower the classifying power. Hence, bi-grams are included in the document token list. Note that here bi-grams are used only with classical machine learning algorithms. Neural network models like recurrent neural networks and transformers that use word embeddings to represent tokens have capabilities to consider the sequence of tokens and hence bi-grams need not be added.

Feature Encoding

At this stage the text documents are in the form of a list of tokens. Machine learning algorithms still cannot directly analyse data in this form and hence the documents need to be encoded into a numeric form as vectors. This process is called vectorization, and the Bag-of-Words(BOW) model is used to represent the vectorized documents. In this model, first a vocabulary is created by considering all unique words in the document collection. Then, each document is represented as a numeric vector as some function of the words present in that document. Each dimension in the vector corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. Two weighting schemes are considered when calculating the value of the term, as follows:

Word frequency weights Using the frequencies of words is the simplest form of weighting words in a document. Each document is represented in terms of the frequency of occurrence of words within the document. Thus, a document consisting of a list of tokens is transformed into a vector consisting of the count of words. Only the word count is retained as a feature and any information about the order of words is discarded.

Term frequency-Inverse document frequency (TF-IDF) weights In this scheme, a document is represented as a vector of term frequency- document frequency (tf-idf) weights of its constituent words/tokens. The tf-idf weight is a statistical measure used to evaluate the importance of a word in a document collection. It consists of two frequencies: the term frequency, and the document frequency. Term Frequency scores how frequently a word appears in a document. Since documents can be of different lengths, it is possible that a word would appear more frequently in longer documents than shorter ones. Hence the word count is adjusted for the document length to arrive at the term frequency.

$$\text{term frequency}, tf = \frac{\text{Number of times word appears in document}}{\text{Document length}}$$

Inverse document frequency scores the rarity of a word across the document collection. Rarer terms have a higher score to reflect their importance. It is the logarithmically scaled inverse fraction of the documents that contain the word.

$$\text{inverse document frequency}, idf = \log \frac{\text{Number of documents}}{\text{Number of documents in which word appears}}$$

$$\text{tf-idf weight} = tf \times idf$$

A high tf–idf weight is reached by a high term frequency (in a given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms and favour more relevant terms.

Selection of top-N words

In chapter 3 it was seen that after pre-processing there were on average 340000 unique tokens in every data-set. Since these were too many features, some dimensionality reduction was needed so that the models learn from the words with the most discriminatory power. This has the advantage of reducing over-fitting on redundant data and improving training efficiency. With reduced number of features, it also helps in interpretability of model behaviour by human stakeholders. The Chi-square feature selection method was first used to determine how many important features were present in general. A brief description of the chi-square statistic is given next.

For selecting features(words) from text data, the χ^2 test is used to test whether the occurrence of a term(word) and the occurrence of a class are independent. Formally, given a document D, we compute the following χ^2 score for each term t and rank the words by their scores to perform the selection:

$$\chi^2(D,t,c) = \sum_{e_t \in \{0,1\}} \sum_{e_c \in \{0,1\}} = \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}}$$

where

- N is the observed frequency in D and E is the expected frequency
- e_t takes the value 1 if the document contains term t and 0 otherwise
- e_c takes the value 1 if the document is in class c and 0 otherwise

For each term (word), a corresponding high χ^2 score indicates that the null hypothesis that the document class has no influence over the word's frequency should be rejected and the occurrence of the word and class are considered dependent. Such dependent words are selected as features. The dependent words are ranked in order of their χ^2 scores and the top-N words are chosen.

This type of feature selection can be viewed as a method for replacing a complex classifier (using all features) with a simpler one (using a subset of the features). It may appear counter-intuitive that a seemingly weaker classifier that uses lesser features is advantageous in text classification, but [Manning et al. 2008] in a detailed discussion of the bias-variance trade-off show that simpler models are often preferable for classifying texts when limited training data are available. Note that the top-N words are used only when using classical machine learning models, and not when using neural networks.

The results after applying all these pre-processing methods are shown described next. Some texts that were very short and did not have any meaningful analytical information were removed. This is shown in the class distribution of the examples after pre=processing in Table 4.1. Table 4.2 shows the text statistics after pre-processing. We note that the corpus size has now reduced by more than 50%. The number of unique tokens have also reduced. This can be attributed to the removal of special characters, single tokens, and stop-words. We also note that the text-token ratio has increased by 80% after pre-processing.

4.2.2 Data Preparation for Deep Learning Models

For the deep transfer learning approach, a pre-trained language model (BERT) will be used. This model was trained to distinguish case differences in words and hence there is no need to convert the text into lower-case characters. This model also has its own tokenizer that processes the text and hence minimal data preparation will need to be done. The pre-processing pipeline that we used for preparing the text data to be analysed by the deep transfer learning based algorithms is illustrated in figure 4.3 and each step is explained next. Similar to the previous section, the preparation starts with removing text records that are too short. The next steps are described further.

Data-set (days before admission)	Number of Positive examples	Number of Negative examples
0 days	186	2200
3 days	183	2200
7 days	183	2200
10 days	182	2200
14 days	182	2200
30 days	181	2200
60 days	179	2200
90 days	179	2200
180 days	171	2200

Table 4.1: Class distributions for all time-frames, after pre-processing

Data-set (days before admission)	Corpus Size (Total Tokens)	Number of Unique Tokens	Type-Token Ratio	Minimum record length (in tokens)	Maximum record length (in tokens)	Mean record length (in tokens)
0 days	21466687	313413	0.01459	21	129944	8996
3 days	21408415	312872	0.01461	21	129944	8983
7 days	21366108	312479	0.01460	21	129944	8966
10 days	21345457	312257	0.01462	21	129944	8961
14 days	21318974	311983	0.01463	21	129944	8950
30 days	21237961	311369	0.01466	21	129944	8919
60 days	21134205	310475	0.01469	21	129944	8883
90 days	21052737	309809	0.01471	21	129944	8849
180 days	20861159	307902	0.01475	21	129944	8798

Table 4.2: Text Statistics for all time-frames, after pre-processing

Removal of Special Characters

Since BERT is a language model, some common punctuation marks used in language must be retained to maintain the natural language structure.

Text Summarization

A constraint imposed by the BERT architecture is a restriction on the size of the maximum length of text that can be used as input. For classification tasks the maximum length of the input can be 512 tokens. Hence, the text in our data-set was to be reduced to this size. There are two major types of text summarization techniques: Extractive and Abstractive. In extractive summarization, key phrases and sentences are selected to make a summary but no new text is generated. In abstractive summarization, new phrases ans sentences are created that convey the most useful information. However, this technique would require a large corpus of training data in order to first develop a summarizer. Hence, we chose the extractive technique to summarize the data. Text summarization based on the TextRank algorithm was used to reduce the length of the input to 500 tokens, in the cases where the text length exceeded this number. If the length was less than 500 tokens, the entire text was retained. The text statistics after summarization are shown in Table 4.3. When compared to the raw text in table 3.2, the type-token ratio has increased by 370%, indicating a higher semantic richness, as a result of the reduction in document length.

Figure 4.3: An overview of the data pre-processing pipeline for deep learning algorithms

Data-set (days before admis-sion)	Total Tokens	Total Types (Unique Tokens)	Type-Token Ratio	Mean Sentence length (in tokens)	Minimum record length (in tokens)	Maximum record length (in tokens)	Mean record length (in tokens)
0 days	1196007	45996	0.03845	18	29	802	501
3 days	1193919	45930	0.03846	18	29	802	501
7 days	1192941	45878	0.03845	18	29	802	500
10 days	1192625	45876	0.03846	18	29	802	500
14 days	1191985	45841	0.03845	18	29	802	500
30 days	1191468	45802	0.03844	18	29	802	500
60 days	1190480	45790	0.03846	18	29	802	500
90 days	1189414	45755	0.03846	18	29	802	500
180 days	1184672	45657	0.03853	18	29	802	499

Table 4.3: Text Statistics for all time-frames, after summarization

4.3 Methodological Frameworks

In this chapter we provide an overview of the various machine learning methodologies that we used for the prediction task. We start with the classical machine learning methodologies in section 4.3.1 and present the deep learning methodology in section 4.3.2. We describe Binary Logistic Regression, Support Vector Machines, and Random Forests in sections 4.3.1, 4.3.1, and 4.3.1 respectively. Then, we present some deep learning preliminaries covering Transformer Networks, Siamese Learning, Transfer Learning, and Bidirectional Encoder Representations from Transformers(BERT) in section 4.3.2, and present the two deep transfer learning frameworks next. We first explain fine-tuning the language model for direct use as a classifier in section 4.3.2 and finally present our proposed Siamese fine-tuning approach for document feature representation in section 4.3.2.

4.3.1 Classical Machine Learning Models

Binary Logistic Regression

Logistic regression is a statistical model that uses a logistic function to model a binary dependent variable. It is a linear classifier that attempts to partition the feature space with a hyper-plane to classify data by optimizing a discriminative objective function. In regression analysis, logistic regression estimates the parameters of a logistic model. Mathematically, a binary logistic model has a dependent variable with two possible values, such as yes/no, which is represented by an indicator variable, where the two values are labeled "0" and "1". The logistic regression model models the probability of output in terms of input and does not perform statistical classification. It can be used to make a classifier, for instance by choosing a cutoff value and classifying inputs with probability greater than the cutoff as one class, and those below the cutoff as the other. Logistic regression performs better on larger documents or data-sets and is very efficient to train [Jurafsky 2000]. However, its performs poorly on non-linear data and is sensitive to noise and highly correlated input features.

Support Vector Machine

Given a set of training examples, each marked as belonging to a particular category, an SVM algorithm builds a model that assigns new examples to one category or the other. The working principle of SVMs is to determine separators in the search space which can best separate different classes. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

More formally, a support-vector machine constructs a hyper-plane or a set of hyper-planes in a high-dimensional space, which can be used for classification or other tasks like regression. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training-data point of any class (functional margin), since in general the larger the margin, the lower the generalization error of the classifier.

Support Vector Machines perform extremely well in classifying high dimensional data and are robust to outliers but require that the classes are linearly separable. Their performance tends to reduce when classes have a large amount of overlap.

Random Forest

Decision trees create class partitions by learning a hierarchical division of the underlying data space with the use of different text features. A text is then classified based on the partition it is most likely to belong to [Aggarwal and Zhai 2012]. A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the data-set and uses averaging to improve the predictive accuracy and control over-fitting. This collection of decision tree classifiers is also known as the forest. Each tree is trained on an independent random sample of the training

data. In a classification problem, each tree votes for a class and the most popular class is chosen as the final result. Significant improvements in classification accuracy have resulted from growing an ensemble of decision trees trained on subsets of the data-set and letting them vote for the most popular class [Breiman 2001]. These classifiers are robust to outliers and generalize well but they need meaningful features to perform well. A large number of noisy features reduces performance.

4.3.2 Deep Transfer Learning

In this section, we first briefly describe the necessary preliminary concepts that are used in the explored models.

Preliminaries

Deep Learning

A deep-learning architecture is a multi-layer stack of simple modules, all (or most) of which are subject to learning, and many of which compute non-linear input–output mappings. Each module in the stack transforms its input to increase both the selectivity and the invariance of the representation. With multiple non-linear layers, say a depth of 5 to 20, a system can implement extremely intricate functions of its inputs that are simultaneously sensitive to minute details — distinguishing sunflowers from lotuses — and insensitive to large irrelevant variations such as the background, pose, lighting and surrounding objects.

Many applications of deep learning use feed-forward neural network architectures, which learn to map a fixed-size input (for example, an image) to a fixed-size output (for example, a probability for each of several categories). To go from one layer to the next, a set of units compute a weighted sum of their inputs from the previous layer and pass the result through a non-linear function. Units that are not in the input or output layer are conventionally called hidden units. The hidden layers can be seen as transforming the input in a non-linear way so that categories become linearly separable by the last layer. Neural networks perform these transformations using parameters that are learnt using the back-propagation algorithm [Goodfellow et al. 2016] along with an appropriate optimizer [Goodfellow et al. 2016].

Deep learning language models use dense vectorized representations for words where each word is associated with a vector of real valued features, and semantically related words have vectors close to each other in that vector space. For language processing, deep-learning theory shows that deep nets have two different exponential advantages over classic learning algorithms that do not use distributed representations. Both of these advantages arise from the power of composition and depend on the underlying data-generating distribution having an appropriate componential structure. First, learning distributed representations enable generalization to new combinations of the values of learned features beyond those seen during training (for example, 2^n combinations are possible with n binary features). Second, composing layers of representation in a deep net brings the potential for another exponential advantage(exponential in the depth) [LeCun et al. 2015].

Transformer Networks

A Transformer network is a feed-forward neural network using only the attention mechanism and is based on the sequence-to-sequence architecture.

The attention-mechanism looks at an input sequence and decides at each step which other parts of the sequence are important. It directly models relationships between all words in a sentence, regardless of their respective position. For example, consider the following sentence “I arrived at the bank after crossing the river”. To determine that the word “bank” refers to the shore of a river and not a financial institution, the Transformer can learn to immediately attend to the word “river” and make this decision in a single step.

A transformer network consists of two main components: a set of encoders chained together and a set of decoders chained together. The function of each encoder is to process its input vectors to generate encodings, which contain information about the parts of the inputs which are relevant to each other. It passes its set of generated encodings to the next encoder as inputs. Each decoder

does the opposite, taking all the encodings and processing them, using their incorporated contextual information to generate an output sequence. For this, each encoder and decoder makes use of an attention mechanism, which for each input, weighs the relevance of every input and draws information from the input accordingly when producing the output. Each decoder also has an additional attention mechanism which draws information from the outputs of previous decoders, before the decoder draws information from the encodings. Both the encoders and decoders have a final feed-forward neural network for additional processing of the outputs, and also contain residual connections and layer normalization steps. Transformers typically undergo semi-supervised learning in a two step approach that involves unsupervised pre-training followed by supervised fine-tuning. Pre-training is typically done on a much larger data-set than fine-tuning, due to the restricted availability of labeled training data.

Siamese Learning

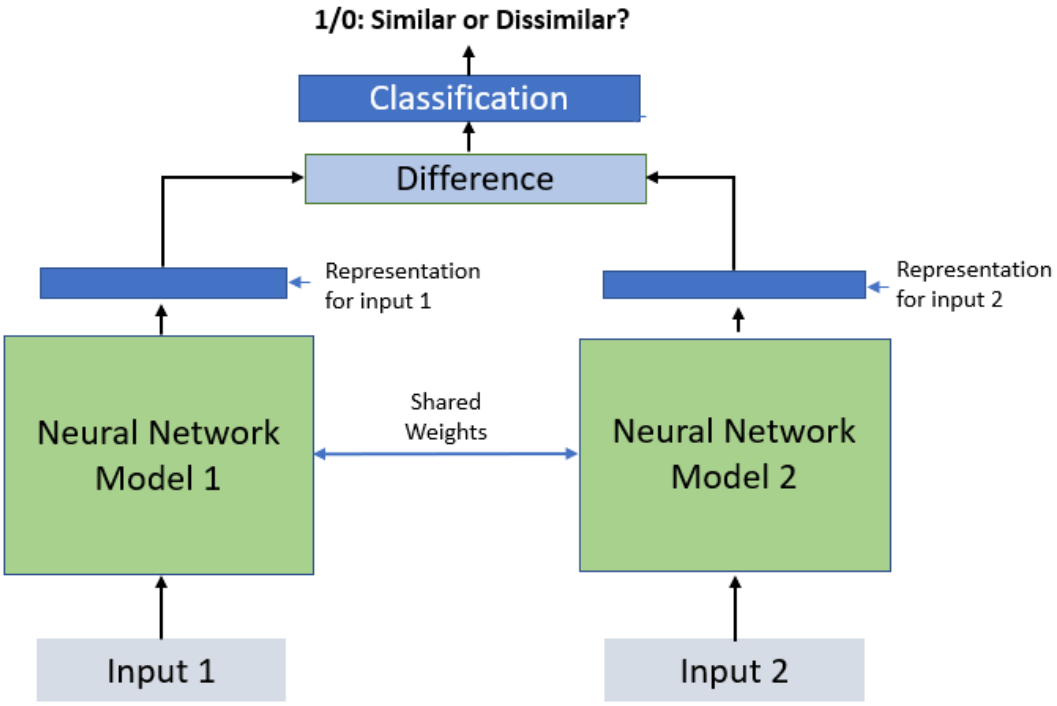

Figure 4.4: Siamese Neural Network Architecture

Siamese Neural Networks (SNNs) are a type of neural networks that have many instances of the same model with shared weights. Since the outputs of these different instances are connected at a later stage, such architectures are referred as Siamese networks similar to how Siamese babies are connected. In this project we make use of SNNs with dual instances of neural networks. In computer vision applications, this architecture shows its strength when it has to learn with limited data such as Zero/One shot learning tasks for image recognition. In this project we investigate whether such SNNs could be used for our data-set. An overview of a twin Siamese architecture is shown in figure 4.4. While traditional neural networks learn to classify data into different classes, SNNs learn to discriminate between different inputs and represent the similarity of the inputs. As shown in the figure, the training data-set for a Siamese network consists of input data pairs (input 1, input 2) and corresponding labels,y. The label y $\in$ {0,1} indicates whether input 1 and input 2 are similar or dissimilar. The aim of training is to minimize the distance between similar

pairs and maximize the distance between dissimilar pairs, in an embedding space. After training, a single instance of the model can then be used as a feature extractor.

We aim to leverage this type of training input structure to build a feature extractor for our data-set. For a small binary data-set with x number of samples belonging to the class of interest, one can artificially create up-to $x*(x-1)/2$ [1] number of similar input pairs. An equivalent number of similar pairs can then be created for the other class. Next, an appropriate number of dissimilar input pairs can be created such that the number of dissimilar pairs is equal to the number of similar input pairs. This process results in a balanced number of similar and dissimilar training examples. The model can then learn discriminative features from this input data.

Transfer Learning

Transfer learning is aimed to make use of valuable knowledge in a source domain to help improve model performance in a target domain. It is particularly important to neural networks, which are very likely to over-fit when trained on small data-sets. Parameters from a network model trained on a source task are transferred to initialize a network on the target task. The model may be trained again using the target data. This is called fine-tuning. The fine-tuned model is then used on the intended target task.

BERT

BERT stands for Bidirectional Encoder Representations from Transformers. It is designed to pre-train deep bidirectional language representations from unlabeled text by jointly conditioning on both left and right context of the input text. It uses the WordPiece [Wu et al. 2016] tokenization algorithm to segment words into sub-words. This type of tokenization is capable of covering a wider spectrum of rare and out-of-vocabulary words since the input text is tokenized at a sub-word level, and this could be very useful in processing our clinical data-set. BERT is pre-trained on a large corpus of unlabelled text including the entire Wikipedia (2,500 million words) and Book Corpus (800 million words). As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of NLP tasks. This BERT model is available as an English model and as a multi-lingual model with support for 104 languages, including Dutch. The multilingual BERT model for Dutch is only based on the Dutch Wikipedia text, which is a specific domain, unrepresentative of general language use, and hence we chose to use a more advanced Dutch BERT model, BERTje [Vries et al. 2019].

BERTje

BERTje is a monolingual Dutch BERT model based on the same architecture as BERT and trained on a large and diverse dataset of 2.4 billion tokens. It was trained using multi-genre data to be more representative of general Dutch language use. In addition to Wikipedia text, this model was also trained on fiction novels, Dutch news articles, and articles from a multi-genre reference corpus. The resulting model has been shown to consistently outperform the multilingual BERT model on downstream NLP tasks like part-of-speech tagging, named-entity recognition and sentiment analysis.

An overview of using BERTje as a classifier model is provided in figure 4.5. When using BERTje as a classifier, it takes the input text as a sequence of at most 512 tokens and outputs the token-wise representation of the sequence. The input sequence has a pre-defined structure where the first token of the sequence is set to be a special [CLS] token. This segment of the output then contains the special classification embedding. For text classification tasks, BERTje takes this final state of the first token [CLS] as the representation of the whole sequence. A softmax layer [Goodfellow et al. 2016] can then be applied at this output to classify the document into classes.

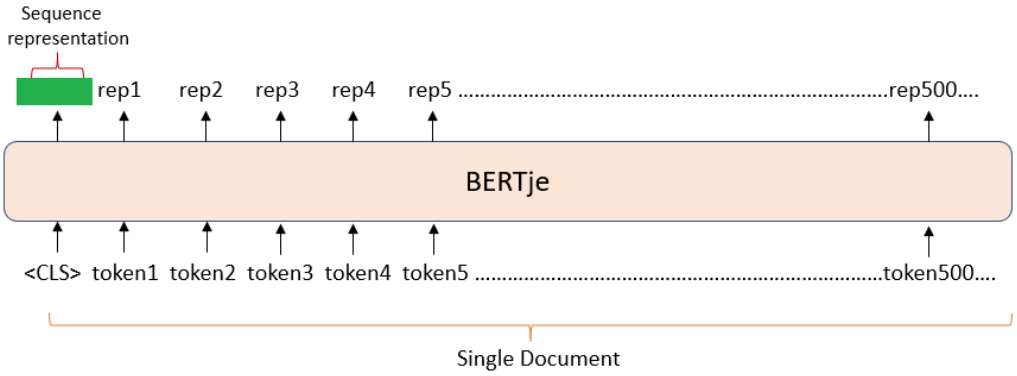

Figure 4.5: BERTje as a classifier

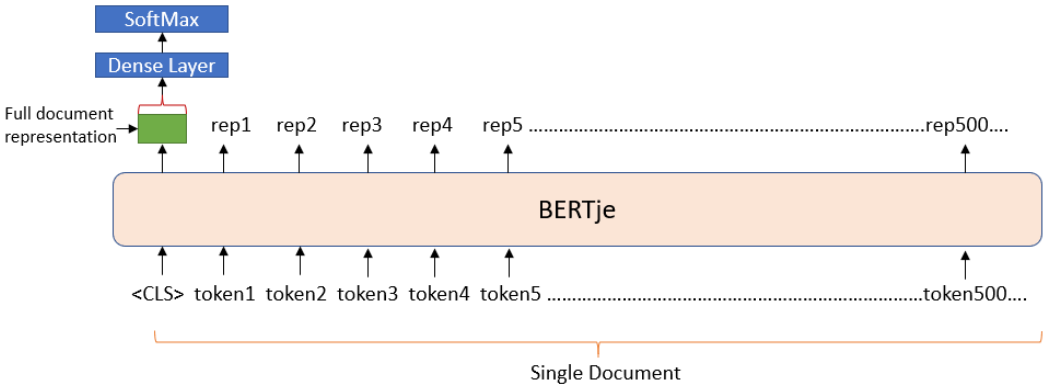

Figure 4.6: Fine-tuning of BERTje as a classifier

Fine-tuning of BERTje for Text Classification

Here we explore using transfer learning to fine-tune BERTje as a classifier. For classification problems, fine-tuning a model as a classifier is the common or regular approach to transfer learning for classification [Sun et al. 2019] and hence we refer to this type of fine-tuning as 'regular' fine-tuning in this work. In Sun et al. [2019], it was shown that fine-tuning on as few as 200 training examples could provide good classification results. Hence, we down-sample the majority class such that it has the same number of examples as the minority class and then use this balanced data-set as the training data-set. Figure 4.6 depicts the model overview when fine-tuning BERTje as a text classifier. As shown, we add a dense layer on top of BERTje, followed by a softmax layer to predict the correct class label. Note that we use only the output from the first <CLS> token that provides a representation for the full input document. We fine-tune all the parameters of BERTje and the parameters for the dense layer jointly by maximising the log-probability of the correct label using the binary cross-entropy loss [Goodfellow et al. 2016].

[1]Each sample can be paired with all other samples. For x samples, the total number of unique pairings possible = (x-1) + (x-2) + (x-3) + ... + 1 = $x * (x-1)/2$

Fine-tuning of BERTje as a Feature Extractor

Here, we use the same classifier model that was trained in section 4.3.2 but instead of classifying the documents, we extract the document representation provided by the output of the <CLS> token, and use the classical ML classifiers to then perform the classification task. This way, we can compare the difference between using the BERTje neural network model as a classifier and a feature extractor. We can also evaluate the quality of document representations by our proposed Siamese fine-tuning approach with respect to the embeddings after regular fine-tuning.

Proposed Siamese Fine-tuning Approach

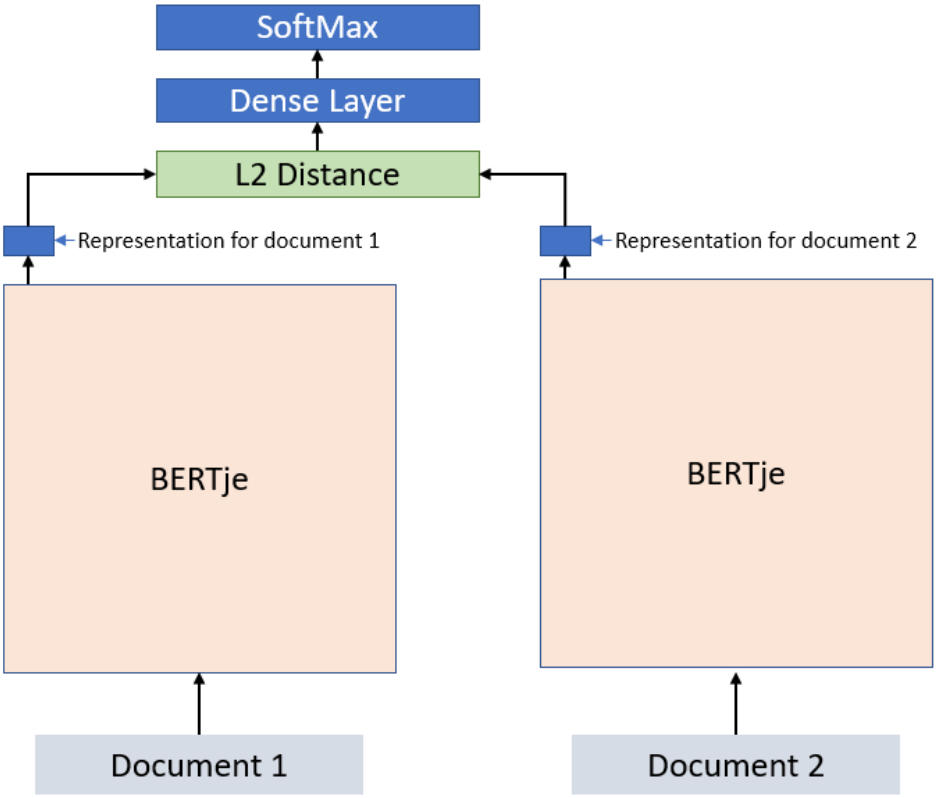

Figure 4.7: Siamese Fine-tuning of BERTje as a feature extractor

Here, we intend to leverage transfer learning to investigate the use of neural network models as feature extractors for our small data-set. The extracted features could then be used in a classical machine learning algorithm. Neural networks are used here as feature extractors to investigate the quality of using dense embeddings as feature representations. In addition to the small size of the data-set, another challenge we face is the highly imbalanced nature of the data-set. Fine-tuning BERTje on a skewed data-set might not be the best approach since it is possible that it would classify all input as belonging to the majority class. Also, fine-tuning models on a down-sampled data-set might also not result in good classifier performance since the model is not trained using all available data. To mitigate this, we propose to use Siamese learning to fine-tune BERTje. We hypothesize that creating a balanced number of positive and negative training samples and then

fine-tuning BERT to learn discriminative features would yield better classification results.

An overview of the fine-tuning architecture is depicted in figure 4.7. The model is setup as 2 instances of the same model with the same weights processing two different inputs. The processed outputs are then merged later to yield the appropriate output. When fine-tuning in this manner, we provide positive and negative samples of document pairs as the dual model input. A positive sample is when the two documents in a pair belong to the same class and a negative sample is when the two documents in a pair belong to different classes. Each document in a pair is input to a BERTje model. The first document is processed by one instance of BERTje and the second document is processed by the second instance of BERTje. Then the L2 distance is calculated between the representations for the two input documents. A dense layer followed by a softmax layer is added to the layer that calculates the L2 distance. All the parameters of BERTje and the parameters for the dense layers are jointly fine-tuned by maximising the log probability of the correct label of the sample pairs. In this architecture, the model learns to minimize the distance between documents representations for documents belonging to the same class and maximizes the distance between document representations for documents belonging to different classes. After fine-tuning, we encode the original text data-set in the form of the document representation provided by the model. These representations can then be used by classical machine learning algorithms capable of handling imbalanced data-sets to perform the required classification.

Chapter 5

Experimental Setup and Results

5.1 Implementation

5.1.1 System Specifications and Programming Tools

All experiments were carried out on a Windows 7 operating system with an Intel Xeon CPU E5-2650 2.0 GHz and 100GB RAM. This was made available through remote access by Antes-Parnassia. The programming language used was Python version 3.7.7. The python libraries numpy and pandas were used for data storage and manipulation. The nltk library was used for implementing the data pre-processing methods. The scikitlearn and scipy libraries were used for implementing the classical machine learning algorithms. The plots were generated using matplotlib. Text summarization was carried out using the gensim library. Lastly, the pytorch and transformers libraries were used to implement the deep transfer learning methods.

5.1.2 Data-sets

As described in chapter 3, 9 data-sets corresponding to 9 different advance prediction time-frames were used. Furthermore, we created two versions of the data-set for training the classifiers corresponding to full and the past month's patient history, as described next.

Full Data

In this version of the data-sets, the entire patient history as provided by Antes was used for training the classifiers.

Last Month Data

Since the full data had the entire patient history of up-to 2 or 3 years and it was possible that only recent data may be more useful in predicting an event such as an involuntary admission, the experiments were performed over data from the last month of the prediction period as well. The last month's text was chosen by selecting only the 9000 most recent characters. This number of characters roughly corresponded to a month's text records. The selected text was then processed in the same manner as the full text. We clarify this selection of the last month's history with the following example.

Consider the data-set 60 days before admission. Suppose a patient A had an involuntary admission on 1st August 2018. This patient would have historical data starting from, for instance, January 2016 up-to 30 April 2018 (all data till 60 days before admission). Now, this entire twenty-eight month history may not be relevant for predicting a sudden event such as an involuntary admission. It is likely that the patient's recent history corresponding to the last month may have more information for the prediction. So in this case, we retain the most recent 9000 characters.

This would approximately correspond to using the data from the month of April 2018 in order to predict whether an admission would occur in August 2018.

It is important to note that this selection does not strictly correspond to a month's history. It could be a longer time-frame for a patient undergoing a less intensive treatment with less frequent observations, or a shorter time-frame for a patient undergoing a more intensive treatment with more frequent observations.

5.1.3 Data Preparation for Classical ML Models

In this section, we describe the particular choices made for the NLP pre-processing methods.

Data Cleaning

Records of length less than 50 characters were removed.

Removal of stopwords

The stop words that were included were: 'niet', 'niets', 'geen', 'maar', and 'zonder'.

Addition of Bigrams

The top 15 bigrams were added to each text record.

Selection of top-N words

Using the chi-square statistic, we observed that for the data-sets on average there were 1700 important (discriminatory) words. Based on this number, it was decided to test different models with the top 100, 250, 500, 750, 1000, 1500, and 2500 chi2 words. The top 2500 words selection was made as a proxy for all words, for performance reasons. Using the full data-set would overload the company's server and hamper entire organizational performance.

5.1.4 Data Preparation for Deep Transfer Learning Models

Removal of Special Characters

All special characters except full-stop(.), question mark(?), comma(,), and exclamation(!) are removed from the text field.

Text Summarization

The text records were summarized to a maximum length of 510 words. In case the record length was already less than 510 characters, it was retained as is.

5.1.5 Modelling Setup

The data was split into training and testing data corresponding to 70% and 30% of the total data. This split was based on recommendations in data mining literature. In addition, this choice corresponded to training the models on 2 year's data and testing on 1 year's data. Such a choice for the testing set is helpful in evaluating the model usability in a practical scenario.

5.1.6 Hyper-parameter Selection and other Modelling Decisions

Classical Machine Learning Models

The hyper-parameters for the three different classifiers were tuned using 5-fold cross validated randomized grid search. For each classifier type, a grid of a range of parameter values was defined

and the best model out of 100 candidate models was selected. The hyper-parameter grid was as follows:

1. Logistic Regression:
    - C: uniform distribution between 0 and 4
    - Penalty: l1, l2
    - Class weights: None, balanced, 1:2, 1:5, 1:10, 1:20
2. Support Vector Machine :
    - C: uniform distribution between 0 and 4
    - Class weights: None, balanced, 1:2, 1:5, 1:10, 1:20
3. Random Forest:
    - Number of trees: 100 to 2000
    - Maximum depth: 15 different depths from the interval (1, 50)
    - Class weights: None, balanced, 1:2, 1:5, 1:10, 1:20

Deep/Transfer Learning Models

Recommendations provided in 'How to fine-tune BERT for Text Classification' [Sun et al. 2019] were used for choosing the fine-tuning hyper-parameters. We varied the recommended hyper-parameters to find a combination that gave quick convergence to a low loss. We noted that the loss tended to converge to a low value by the 2nd or 3rd epoch and increased thereafter. Hence, the models were fine-tuned for 4 epochs and the best model was selected based on the model with the lowest loss on the validation data-set. Since our data-set was already small, we chose to use a small batch size of 16 training examples. The final hyper-parameters used for both, the regular fine-tuning and Siamese fine-tuning were as follows:

- Number of epochs: 4
- Batch size: 16
- Learning rate: 5e-5

Apart from these hyper-parameters, we used a Dense layer of 768 neurons followed by a SoftMax layer with 2 outputs for both the types of fine-tuning architectures. The parameters of the Dense layer provide the model with some more capability to learn to perform the classification. The SoftMax layer with 2 outputs is used since this is a binary classification. Next, we used the cross entropy loss [Goodfellow et al. 2016] function along with the Adam optimizer [Kingma and Ba 2014] for back-propagation as recommended in Sun et al. [2019] for fine-tuning the models.

In addition, for the Siamese fine-tuning, we used a data-set with 1200 training examples. The examples were organized as 600 pairs of positive samples where the text samples in a training pair were from the same class and 600 pairs of negative samples, where the text samples in a training pair were from different classes. There was an operational constraint when fine-tuning in a Siamese manner. The fine-tuning could not be performed over the day-time since the high CPU and RAM utilization would hamper the work of other users in the organization. Hence a model for one data-set could be fine-tuned over one night only. Thus, this value of 1200 training examples was chosen such that this time constraint could be satisfied.

5.1.7 Evaluation Metrics

Since we were working with a highly imbalanced data-set, accuracy was not used as a model performance metric. Precision, recall, F1 score, Area under the Precision-Recall curve, and Area under the kappa curve are reported as model performance measures. All these metrics are briefly described below.

- Precision: The percentage of the model predictions that were correctly classified.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- Recall: The percentage of total correct results that were correctly classified.

$$\text{Recall} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

- F1 score: The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. It is given by

$$F1 = 2 * (precision * recall)/(precision + recall)$$

- Area under Precision-Recall curve (PR-AUC): It is a plot of the precision of the classifier as a function of its recall. In imbalanced data-sets, there are a high number of true negatives and a lower number of true positives, and the goal is usually to obtain a classifier that is good at discriminating these positive examples. Since Precision and Recall are both not dependent on the true negatives, using the area under the Precision-Recall curve helps one evaluate the performance of a classifier with respect to how well it can classify the positive examples. We determine a baseline value for PR-AUC as follows: We note that irrespective of the recall obtained, the best precision one can obtain by random guessing is the real fraction of positive samples in the data-set. Thus, for our data-set this value is on average, $181/2200 = 0.08$. Classifiers with PR-AUC values less than or equal to this baseline can be considered random classifiers with no skill.

- Area under Kappa curve (AUK): The AUK [Kaymak et al. 2010] is a measure of classifier performance, useful in evaluating classifiers for skewed data-sets. Here Cohen's kappa values are plotted against the false positive rates at different classifier thresholds. In the classification setting, Cohen's kappa is used to evaluate the agreement between the ground truth and the predictions made by a classifier, while taking into account correct predictions by chance. The area under this curve thus gives another measure of overall performance of the classifier. Positive values for AUK would denote skillful classifiers.

The confusion metrics (true positives, true negatives, false positives, and false negatives) were also used where appropriate. These help one evaluate the model use in a practical setting. In addition, these also help differentiate between a model with and without skill. For imbalanced data-sets, a model that classifies all examples as the majority class would be considered as having no skill. It must be noted that this is only a preliminary analysis and we provide the confusion metrics for the default setting. In a real world setting, such models would be ideally calibrated to suit the purpose of the institution or company deploying the models.

5.2 Results

In this section we present the results of our experiments for the various modelling methodologies. We first begin by presenting the results of training the classical ML algorithms on the data with various input feature representations (BOW/Tf-idf, Top-N chi^2 words). Next we present the

results of fine-tuning BERTje as a classifier. Then we evaluate whether classical machine learning classifiers can classify the embeddings provided by this model. Next we provide the results of using classical machine learning classifiers to classify the embeddings after Siamese fine-tuning. For all the modelling methodologies, we present model quality metrics and confusion metrics for the best models, and discuss our findings.

5.2.1 Classical Machine Learning: Best Models

In this section, we elaborate over our findings for the experiments using NLP techniques along with classical machine learning classifiers for the prediction task. The results for all the experiments using various input representations are provided in Appendix C. The configurations for the best models for full and past month data are provided in Tables 5.1 and 5.2 respectively. We note that the logistic regression and support vector machine classifiers use some degree of class weights for the under-represented class. The random forest almost always prefers balanced class weights. There were some random forest models that did not use any class weights. Next, we note that the random forest classifier always performs well using the tf-idf form of input representation, while the logistic regression and support vector machine classifiers use either the BOW of tf-idf representations. Further, we observe that all classifiers tend to perform better using 100 to 250 input features. This indicates that even though there are around 310000 unique tokens in each of the data-sets, only a small number of these tokens are important for classification. The best random forest model almost always has a large number of trees with a maximum depth of 8 along with balanced class weighting. The small value of the maximum depth again indicates that only a few features were sufficient for the classification task. Further, the preference for l1 penalty by the logistic regression classifier shows that there are some words that were representative of the class of interest. l1 penalty favors sparsity of features, in contrast to l2 penalty. Sparse input features indicate that only some features have a non-zero value, which are used to discriminate one class from the other.

The model quality metrics for the models using the full and past month data are provided in tables 5.3 and 5.4 respectively. We note that all the models perform better than random guessing. We note this by the fact that for these models, the PR-AUC is greater than the baseline of 0.08, along with a positive AUK value. We observe that the predictive power diminishes as the prediction time-frames move farther from the time of involuntary admission. We note that the F1 score, PR-AUC and AUK are greater for models trained on the past month's patient. Thus, models are able to perform much better when using the last one month's patient history rather than the full patient history, especially for predicting 0 and 3 days in advance. This difference in model quality after using the full and the past month's patient history for the logistic regression classifier is illustrated in Precision-Recall plots in figures 5.1 and 5.2 respectively. Note that the area under the PR curve is larger for the models trained on the past month's data, especially for predicting 0 and 3 days in advance. Similar plots for the other classifiers are provided in Appendix B. To evaluate the suitability of model use in a practical setting, we refer to the confusion metrics shown in tables 5.5 and 5.6 for models using full and past month's data, respectively. Here, we observe that using the previous month's data results in models with better predictive power. The models predicting up-to 0 and 3 days in advance tend to have a high number of true positives along with a low number of false positives, resulting in models with a true positive rate of around 40%. The models for the other time-frames have a very high number of false positives, making them unsuitable for a practical setting.

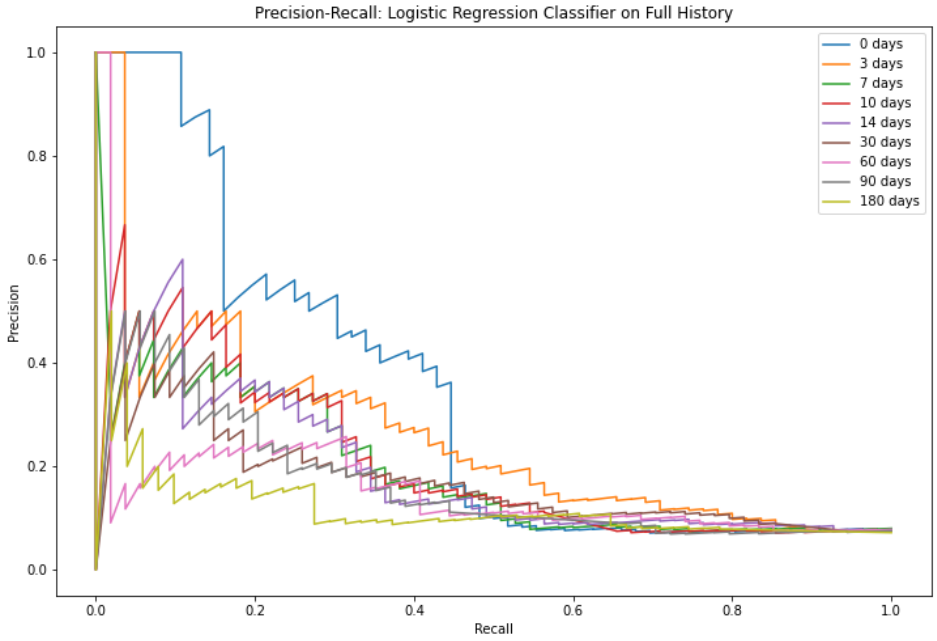

Figure 5.1: Precision-Recall Curves for Logistic Regression Classifiers (Full History)

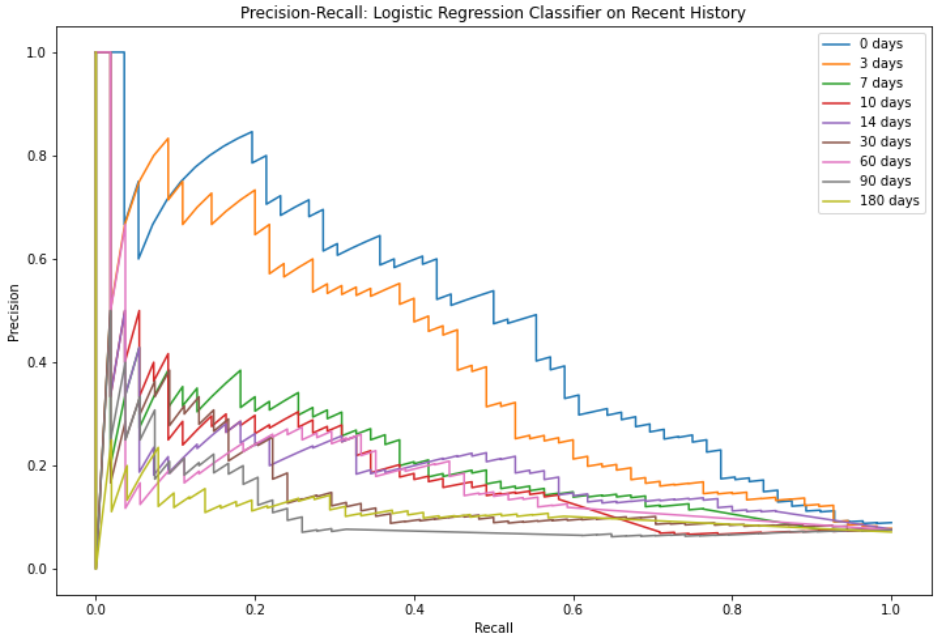

Figure 5.2: Precision-Recall Curves for Logistic Regression Classifiers (Past Month History)

Dataset (Days before admission)	LR	SVC	RF
0	C:0.0005 , class weights:(1:5) , penalty: l2, Top-250 BOW	C: 2.74, class weights:(1:5) Top-100 Tf-idf	estimators: 2000, max depth: 8 , class weights: balanced, Top-100 BOW
3	C: 1.16 , class weights: (1:5), penalty: l1, Top-100 Tf-idf	C: 0.04 , class weights:balanced Top-250 BOW	estimators: 2000, max depth: 8 , class weights: balanced, Top-500 Tf-idf
7	C: 1.16, class weights: (1:5), penalty: l1, Top-250 BOW	C: 0.2 , class weights: (1:10) Top-250 Tf-idf	estimators:2000 max depth: 8, class weights: balanced, Top-750 Tf-idf
10	C: 0.0005, class weights: (1:5), penalty: l2, Top-100 Tf-idf	C: 0.29 , class weights: balanced Top-250 BOW	estimators: 2000 , max depth: 8, class weights: balanced, Top-250 Tf-idf
14	C: 1.16, class weights: (1:5), penalty: l1, Top-250 BOW	C: 0.07 , class weights: (1:5), Top-250 Tf-idf	estimators: 2000 , max depth: 8 , class weights: balanced Top-250 Tf-idf
30	C: 1.06, class weights: balanced, penalty: l1, Top-250 Tf-idf	C: 0.29, class weights:balanced Top-250 Tf-idf	estimators:2000 , max depth:8 , class weights:balanced Top-250 Tf-idf
60	C:1.16 , class weights: (1:5), penalty: l1, Top-100 BOW	C: 0.29, class weights:balanced Top-250 BOW	estimators: 2000, max depth:8 , class weights:balanced Top-500 Tf-idf
90	C: 0.0005, class weights:(1:5) , penalty: l2, Top-250 Tf-idf	C: 1.16, class weights: (1:5) Top-250 BOW	estimators:2000 , max depth:8 , class weights:balanced Top-500 Tf-idf
180	C:1.06 , class weights: balanced, penalty: l1, Top-250 Tf-idf	C:2.46 , class weights:(1:10) Top-100 BOW	estimators:2000 , max depth:8 , class weights:balanced Top-100 Tf-idf

Table 5.1: Model Parameters: Best Classical Machine Learning Models on Full History

Dataset (Days before admission)	LR	SVC	RF
0	C: 0.33, class weights: none, penalty: l1, Top-100 BOW	C: 0.11 , class weights: (1:2), Top-100 BOW	estimators: 1788 , max depth:25 , class weights: none, Top-100 Tf-idf
3	C: 0.0005 , class weights: (1:5), penalty: l2, Top-250 BOW	C: 0.11, class weights: (1:2), Top-250 BOW	estimators:2000 , max depth: 8, class weights: balanced, Top-100 Tf-idf
7	C: 1.16 , class weights: (1:5), penalty: l1, Top-100 Tf-idf	C:0.11 , class weights:(1:5), Top-100 Tf-idf	estimators:2000 , max depth:8 , class weights: balanced, Top-100 Tf-idf
10	C: 0.0005, class weights: (1:5), penalty: l2, Top-100 Tf-idf	C: 3, class weights: (1:2), Top-100 BOW	estimators: 944 , max depth: 36 , class weights: none, Top-250 Tf-idf
14	C: 1.16, class weights: (1:5), penalty: l1, Top-100 Tf-idf	C:0.34 , class weights: (1:10), Top-100 BOW	estimators: 100 , max depth: 32, class weights: none, Top-250 Tf-idf
30	C: 2.8, class weights: balanced, penalty: l1, Top-2500 BOW	C:0.35 , class weights:none, Top-250 BOW	estimators:2000 , max depth: 8, class weights:balanced, Top-100 Tf-idf
60	C:1.06 , class weights:balanced , penalty: l1, Top-100 Tf-idf	C:3.39 , class weights:(1:20), Top-100 Tf-idf	estimators:2000 , max depth:8 , class weights:balanced Top-500 Tf-idf
90	C:0.0005 , class weights:(1:5), penalty: l2, Top-250 Tf-idf	C:0.29 , class weights:balanced, Top-100 Tf-idf	estimators:2000 , max depth: 8, class weights:balanced Top-250 Tf-idf
180	C: 1.67, class weights: (1:10), penalty: l1, Top-250 Tf-idf	C: 3.73, class weights:balanced Top-100 Tf-idf	estimators: 2000, max depth:8 , class weights:balanced Top-100 Tf-idf

Table 5.2: Model Parameters: Best Classical Machine Learning Models on Past Month History

Dataset (Days)	LR			SVC			RF		
	F1	PR-AUC	AUK	F1	PR-AUC	AUK	F1	PR-AUC	AUK
0	0.34	0.33	0.07	0.35	0.37	0.12	0.29	0.33	0.10
3	0.32	0.25	0.09	0.31	0.16	0.06	0.33	0.34	0.12
7	0.30	0.19	0.06	0.27	0.20	0.08	0.21	0.26	0.10
10	0.29	0.20	0.06	0.27	0.21	0.07	0.27	0.27	0.10
14	0.29	0.18	0.06	0.28	0.22	0.09	0.29	0.26	0.10
30	0.24	0.17	0.07	0.26	0.18	0.07	0.19	0.17	0.08
60	0.21	0.24	0.06	0.26	0.16	0.07	0.23	0.17	0.07
90	0.21	0.16	0.05	0.27	0.18	0.07	0.22	0.16	0.07
180	0.19	0.11	0.04	0.20	0.14	0.05	0.15	0.11	0.04

Table 5.3: Model Quality metrics: Classical Machine Learning Models (Full History)

Dataset (Days)	LR			SVC			RF		
	F1	PR-AUC	AUK	F1	PR-AUC	AUK	F1	PR-AUC	AUK
0	0.48	0.47	0.15	0.44	0.42	0.13	0.46	0.48	0.14
3	0.44	0.39	0.13	0.38	0.33	0.10	0.42	0.39	0.12
7	0.31	0.20	0.09	0.28	0.22	0.09	0.19	0.21	0.09
10	0.29	0.18	0.07	0.27	0.19	0.07	0.16	0.22	0.09
14	0.23	0.20	0.09	0.24	0.17	0.06	0.13	0.20	0.08
30	0.19	0.24	0.05	0.18	0.15	0.04	0.15	0.12	0.05
60	0.26	0.16	0.07	0.28	0.18	0.08	0.21	0.18	0.06
90	0.18	0.11	0.01	0.26	0.19	0.07	0.24	0.15	0.06
180	0.15	0.11	0.04	0.19	0.12	0.04	0.16	0.10	0.04

Table 5.4: Model Quality metrics: Classical Machine Learning Models (Past Month History)

Dataset (days before admission)	LR				SVC				RF			
	TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
0	14	12	648	42	16	19	641	40	11	9	651	45
3	18	38	622	37	21	61	599	34	12	5	655	43
7	16	34	626	39	22	86	574	33	7	4	656	48
10	14	28	632	41	20	71	589	35	11	16	644	44
14	17	45	615	38	14	32	628	41	12	16	644	43
30	20	92	569	34	22	95	566	32	9	31	630	45
60	10	32	628	44	19	74	586	35	9	15	645	45
90	9	22	638	45	12	23	637	42	8	11	649	46
180	14	82	579	37	10	38	623	41	7	35	626	44

Table 5.5: Confusion metrics: Classical Machine Learning Classifiers (Full History)

Dataset (days before admission)	LR				SVC				RF			
	TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
0	23	16	644	33	22	22	638	34	18	4	656	38
3	22	23	637	33	20	29	631	35	17	8	652	38
7	17	39	621	38	14	30	630	41	8	21	639	47
10	18	53	607	37	12	22	638	43	5	4	656	50
14	12	36	624	43	16	64	596	39	4	2	658	51
30	8	22	639	46	8	28	633	46	7	32	629	47
60	19	74	586	35	21	77	583	33	8	15	645	46
90	10	47	613	44	18	68	592	36	11	26	634	43
180	10	74	587	41	15	89	572	36	10	68	593	41

Table 5.6: Confusion metrics: Classical Machine Learning Classifiers (Past Month History)

5.2.2 Deep/Transfer Learning: Best Models

Regular Fine-tuning for Classification

In this section, we discuss the results of fine-tuning BERTje as a classifier. The model quality and confusion metrics for models trained using the full and the previous month's patient history are provided in tables 5.7 and 5.8 respectively. We observe that the PR-AUC is just marginally above the baseline of 0.08 for most of the time-frames. This classifier has a performance better than random guessing only when predicting 0 days in advance using the previous month's patient history. When observing the confusion metrics, although this model has a true positive rate of 60%, the very high number of false positives make it unsuitable for practical use. We also see in the confusion metrics that all other models tend to classify all test examples as belonging to one of the two classes, reflecting the poor model quality.

Data-set (Days before admission)	F1	PR-AUC	AUK	TP	FP	TN	FN
0	0.15	0.08	-0.00	56	660	0	0
3	0.00	0.11	0.02	55	660	0	0
7	0.14	0.14	0.05	55	660	0	0
10	0.20	0.14	0.06	38	287	373	17
14	0.00	0.08	-8.72	0	0	660	55
30	0.14	0.17	0.06	54	660	0	0
60	0.00	0.14	0.07	0	2	658	54
90	0.14	0.06	-0.03	54	660	0	0
180	0.13	0.08	0.01	51	661	0	0

Table 5.7: Model quality and confusion metrics: BERTje as a Classifier (Full History)

Data-set (Days before admission)	F1	PR-AUC	AUK	TP	FP	TN	FN
0	0.28	0.27	0.10	34	150	510	22
3	0.14	0.11	0.05	55	660	0	0
7	0.00	0.13	0.04	0	0	660	55
10	0.00	0.06	-0.04	0	0	660	55
14	0.14	0.10	0.03	55	660	0	0
30	0.14	0.09	0.02	54	661	0	0
60	0.14	0.10	0.04	54	660	0	0
90	0.14	0.09	0.02	54	660	0	0
180	0.13	0.10	-0.00	51	661	0	0

Table 5.8: Model quality and confusion metrics: BERTje as a Classifier (Past Month histroy)

Regular Fine-tuning for feature extraction

In this section, we discuss the results of extracting document embeddings from a BERtje model that was fine-tuned for classification, and using the classical machine learning algorithms to perform the classification. Logistic Regression (LR), Support Vector Classifier (SVC), and Random Forests (RF) are used as the classifiers. The parameters for the best classifiers are provided in Appendix D. We observe that all models used class weights for the under-represented class. The model quality metrics for models fine-tuned on the full and the previous month's patient history are provided in tables 5.9 and 5.10 respectively. We observe that model performance diminishes as the prediction time-frames move farther from the time of involuntary admission, and models trained on the previous month's patient history have better predictive power when compared to models trained on full patient history. We note that the PR-AUC is greater than the baseline of 0.08 when predicting at time-frames closer to the time of involuntary admission and is only marginally greater than the baseline for time-frames such as 60, 90, and 180 days in advance. We also note that the random forest classifier has the least predictive power. The confusion metrics for models fine-tuned on full and the previous month's patient history are provided in tables 5.11 and 5.12 respectively. Here, we clearly see the low predictive power of the random forest since it classified most examples as belonging to the majority class. For the logistic regression and support vector machine, though they do have some predictive power, the high number of false positives tends to make these models unsuitable for practical use.

We note that though the classifier model had a poor performance, the embeddings generated by the model still had some information that allowed other classifiers to classify the examples. We also note that the poor performance of the random forest classifier could indicate that the individual dimensions of the document embeddings cannot be partitioned in a hierarchical manner.

Dataset (Days before admission)	LR			SVC			RF		
	F1	PR-AUC	AUK	F1	PR-AUC	AUK	F1	PR-AUC	AUK
0	0.18	0.12	0.05	0.19	0.13	0.05	0.00	0.14	0.06
3	0.14	0.13	0.03	0.14	0.13	0.03	0.03	0.12	0.04
7	0.09	0.09	0.01	0.10	0.10	0.01	0.00	0.11	0.04
10	0.22	0.19	0.07	0.22	0.17	0.06	0.11	0.15	0.07
14	0.16	0.12	0.04	0.16	0.11	0.03	0.00	0.09	0.00
30	0.24	0.27	0.08	0.27	0.24	0.08	0.04	0.23	0.08
60	0.12	0.08	0.01	0.12	0.08	0.01	0.00	0.10	0.03
90	0.09	0.10	0.01	0.14	0.08	0.01	0.00	0.08	0.01
180	0.10	0.09	0.02	0.08	0.07	0.01	0.04	0.10	0.02

Table 5.9: Model quality metrics: Classical ML Classifiers on Embeddings after Regular Fine-tuning (Full History)

Dataset (Days before admission)	LR			SVC			RF		
	F1	PR-AUC	AUK	F1	PR-AUC	AUK	F1	PR-AUC	AUK
0	0.33	0.36	0.11	0.35	0.34	0.10	0.22	0.32	0.12
3	0.31	0.22	0.09	0.22	0.19	0.07	0.16	0.26	0.09
7	0.27	0.21	0.08	0.27	0.23	0.08	0.06	0.15	0.06
10	0.21	0.18	0.07	0.24	0.17	0.08	0.07	0.18	0.07
14	0.18	0.14	0.06	0.13	0.11	0.02	0.00	0.14	0.04
30	0.16	0.16	0.04	0.26	0.13	0.05	0.00	0.11	0.03
60	0.15	0.10	0.03	0.11	0.09	0.03	0.00	0.10	0.02
90	0.14	0.11	0.02	0.15	0.11	0.03	0.00	0.10	0.02
180	0.12	0.09	0.03	0.13	0.11	0.03	0.00	-0.01	0.10

Table 5.10: Model quality metrics: Classical ML Classifiers on Embeddings after Regular Fine-tuning (Past Month History)

Dataset (days before admission)	LR				SVC				RF			
	TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
0	19	136	524	37	11	50	610	45	0	0	660	56
3	14	131	529	41	13	124	536	42	1	3	657	54
7	6	69	591	49	9	110	550	46	0	0	660	55
10	31	198	462	24	18	93	567	37	4	14	646	51
14	20	176	484	35	20	170	490	35	0	0	660	55
30	10	18	642	44	14	37	623	40	1	0	660	53
60	11	123	537	43	11	116	544	43	0	1	659	54
90	8	111	549	46	17	165	495	37	0	0	660	54
180	11	152	508	40	7	110	550	44	1	3	657	50

Table 5.11: Confusion metrics: Classical ML Classifiers on Embeddings after Regular Fine-tuning (Full History)

Dataset (days before admission)	LR				SVC				RF			
	TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
0	17	30	630	39	20	38	622	36	8	10	650	48
3	15	26	634	40	19	96	564	36	5	2	658	50
7	15	41	619	40	24	98	562	31	2	6	654	53
10	13	54	606	42	27	143	517	28	2	1	659	53
14	25	195	465	30	9	71	589	46	0	1	659	55
30	22	192	469	32	16	51	610	38	0	0	661	54
60	17	156	504	37	8	89	571	46	0	2	658	54
90	16	166	494	38	15	127	533	39	0	0	660	54
180	10	105	556	41	11	112	549	40	0	2	659	51

Table 5.12: Confusion metrics: Classical ML classifiers on Embeddings after Regular Fine-tuning (Past Month History)

Siamese Fine-tuning

In this section, we discuss the results of extracting document embeddings extracted from a BERtje model that was fine-tuned using Siamese learning, and using the classical machine learning algorithms (Logistic Regression (LR), Support Vector Classifier (SVC), and Random Forests (RF)) to perform the classification. The parameters for the best classifiers are provided in Appendix D. Again, we observe that all models used class weights for the under-represented class. The model quality metrics when fine-tuned on full and the previous month's patient history are provided in tables 5.13 and 5.14. Similar to models in previous sections, model performance diminishes as the prediction time-frames move farther from the time of involuntary admission, and models trained on the previous month's patient history have better predictive power when compared to models trained on full patient history. We note that the PR-AUC is greater than the baseline of 0.08 when predicting at time-frames closer to the time of involuntary admission and is only marginally greater than the baseline for time-frames such as 60, 90, and 180 days in advance. We also note that the random forest classifier has the least predictive power in terms of model quality metrics. The confusion metrics for models fine-tuned on full and the previous month's patient history are provided in tables 5.15 and 5.16 respectively. Here, we clearly see the low predictive power of the random forest since it classified most examples as belonging to the majority class. For the logistic regression and support vector machine, though they do have some predictive power, the high number of false positives tends to make these models unsuitable for practical use yet.

Dataset (Days before admission)	LR			SVC			RF		
	F1	PR-AUC	AUK	F1	PR-AUC	AUK	F1	PR-AUC	AUK
0	0.19	0.16	0.07	0.20	0.14	0.06	0.00	0.10	0.02
3	0.18	0.12	0.03	0.15	0.11	0.03	0.00	0.07	-0.01
7	0.15	0.10	0.03	0.15	0.09	0.02	0.00	0.08	0.01
10	0.16	0.11	0.03	0.17	0.11	0.03	0.00	0.09	0.01
14	0.17	0.10	0.03	0.17	0.10	0.04	0.00	0.09	0.02
30	0.14	0.15	0.03	0.14	0.12	0.03	0.00	0.08	0.01
60	0.19	0.12	0.05	0.17	0.11	0.04	0.00	0.07	-0.01
90	0.18	0.10	0.03	0.15	0.10	0.03	0.00	0.08	0.01
180	0.08	0.07	-0.00	0.10	0.07	-0.01	0.00	0.08	0.01

Table 5.13: Model quality metrics: Classical ML Classifiers on Embeddings after Siamese Fine-tuning (Full History)

Dataset (Days before admission)	LR			SVC			RF		
	F1	PR-AUC	AUK	F1	PR-AUC	AUK	F1	PR-AUC	AUK
0	0.30	0.20	0.09	0.27	0.20	0.09	0.09	0.14	0.04
3	0.31	0.26	0.09	0.14	0.26	0.09	0.00	0.17	0.05
7	0.14	0.09	0.02	0.19	0.12	0.04	0.00	0.09	0.02
10	0.24	0.13	0.06	0.22	0.13	0.06	0.05	0.11	0.04
14	0.16	0.15	0.04	0.19	0.12	0.04	0.00	0.12	0.05
30	0.15	0.09	0.02	0.17	0.09	0.02	0.00	0.07	0.01
60	0.17	0.11	0.03	0.17	0.10	0.03	0.00	0.08	0.01
90	0.11	0.09	0.01	0.12	0.09	0.01	0.00	0.08	0.01
180	0.15	0.13	0.04	0.19	0.12	0.05	0.00	0.09	0.02

Table 5.14: Model quality metrics: Classical ML Classifiers on Embeddings after Siamese Fine-tuning (Past Month History)

Dataset (days before admission)	LR				SVC				RF			
	TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
0	10	37	623	46	18	105	555	38	0	3	657	56
3	20	142	518	35	17	154	506	38	0	0	660	55
7	19	175	485	36	19	188	472	36	0	0	660	55
10	20	169	491	35	24	199	461	31	0	0	660	55
14	26	219	441	29	24	196	464	31	0	0	660	55
30	9	69	591	45	17	179	481	37	0	0	660	54
60	22	153	507	32	21	166	494	33	0	2	658	54
90	23	185	475	31	14	122	538	40	0	0	660	54
180	8	134	526	43	11	165	495	40	0	0	660	51

Table 5.15: Confusion metrics: Classical ML Classifiers on Embeddings after Siamese Fine-tuning (Full History)

Dataset (days before admission)	LR				SVC				RF			
	TP	FP	TN	FN	TP	FP	TN	FN	TP	FP	TN	FN
0	19	52	608	37	18	61	599	38	3	11	649	53
3	21	58	602	34	5	9	651	50	0	0	660	55
7	16	163	497	39	16	99	561	39	0	0	660	55
10	28	152	508	27	26	160	500	29	2	16	644	53
14	17	140	520	38	23	165	495	32	0	7	653	55
30	16	143	518	38	19	157	504	35	0	2	659	54
60	16	115	545	38	15	107	553	39	0	2	658	54
90	10	117	543	44	9	91	569	45	0	0	660	54
180	15	139	522	36	26	193	468	25	0	1	660	51

Table 5.16: Confusion metrics: Classical ML Classifiers on Embeddings after Siamese Fine-tuning (Past Month History)

5.3 Discussion

5.3.1 Full vs. Past Month Patient History

We note that using recent patient history to perform the analysis results in better models when using both, the classical machine learning algorithms, and the deep transfer learning based models. The model quality metrics F1, PR-AUC and AUK are all greater when models are trained on recent patient history as opposed to full patient history. This difference is also visually illustrated by the precision-recall plots.

5.3.2 Classical Machine Learning Classifiers

We observe that the best classical models tend to favor a lower number of features (top 250 or 100 words) along with some degree of weighting for the minority class. The logistic regression and support vector classifier use both, the BOW and Tf-idf input representations; while the random forest classifier performs well using only the Tf-idf input representation.

5.3.3 Classical Machine Learning Classifier vs. Deep Transfer Learning Classifier

We note that the classical machine learning algorithms result in much better models in terms of model quality and confusion metrics when compared to deep learning based classifiers. The deep learning based classifier exhibits no skill at classification. The PR-AUC is just marginally greater

than baseline and based on the confusion metrics, it classifies all testing samples as belonging to just one of the classes for a large majority of the data-sets. It also exhibits no skill when fine-tuned on the recent patient history. This deep transfer learning based model is recommended in literature and is shown to have state-of-the-art classification power. It is interesting to note that this model had the worst skill in our experiments.

5.3.4 Classical Machine Learning Classifiers on Embeddings after Fine-tuning

We note that when classical ML algorithms are used to classify the embeddings generated by a fine-tuned model, the resulting models have some classification skill. This is seen when embeddings generated after both, regular and Siamese fine-tuning, are used. In these experiments, we observe that while logistic regression and support vector machines were able to classify the document embeddings, random forests were unable to do. Random forests tended to classify all testing samples as belonging to the majority class for all the data-sets, regardless of whether full or recent history was used for fine-tuning the model.

Comparing the model quality metrics, we note that the performance of the classifiers trained on the embeddings on the regularly fine-tuned models was slightly higher. When we observe the confusion metrics, we note that the models trained on the regularly fine-tuned embeddings tend to result in fewer false positives when compared to the models trained on the embeddings after Siamese fine-tuning. Additionally, the classifiers trained on the Siamese embeddings have slightly higher true positives and lower false negatives when compared to the models based on regular fine-tuning. However, for classifiers based on both types of embeddings, the false positives are too high to be of use in a practical setting.

For our small and imbalanced data-set with 181 examples for the under-represented positive class of interest, it was theoretically possible to create up-to $180*179/2 = 16110$ pairs of positive samples pairs. However, in practice, we were able to use only 300 samples due to operational constraints. In computer vision, the number of positive pairs is usually 5 to 10 times the number of positive samples. Since in our experiments the number of pairs is just 1.5 times that of the number of positive examples, we believe this to be a major reason for the lower performance. There is a strong possibility that performance could increase on experiments using a larger number of paired samples. Hence, it is premature to conclude with certainty that the Siamese fine-tuning approach has a poor performance compared to the regular fine-tuning approach.

5.3.5 Feasible Advance Prediction Time-frames

From our experiments, we observe that advance prediction for all time-frames may not be feasible yet. For practical settings, we observe that only for the time-frames 0 and 3 days before admission the classical machine learning models may be used on recent patient history; with at least 0.39 for PR-AUC, F1 scores around 0.44, and a true positive rate around 40%. In such settings, the number of false positives is still low enough for such a classifier to be of some benefit to a company. For all other time-frames, however, though the classifiers have some skill, the number of false positives is too high to be useful in a practical setting.

While the classifiers trained on the deep transfer learning based embeddings are able to reach a true positive rate of upto 56% (e.g., logistic regression classifier trained on embeddings after regular fine-tuning on full patient history 10 days before admission, table 5.11), similar to the classical machine learning models, the very high number of false positives makes such models unsuitable for practical use.

Chapter 6

Conclusions

6.1 Discussion

We explored classical machine learning and deep transfer learning models in order to predict involuntary psychiatric admissions using Dutch EHR data provided by Antes Groep. We trained various models to investigate whether involuntary admissions can be predicted days or even months in advance. We report the F1 scores, Area under Kappa curve, and Area under Precision-Recall, and confusion metrics. The performance of our models is mainly evaluated using area under the precision-recall curve (PR-AUC), and we use the confusion metrics to evaluate model use in practical settings. The PR-AUC is greater than 0.35 for the classical ML classifiers and greater than 0.20 for the deep transfer learning based classifiers when predicting up-to 3 days in advance using the last month's patient history. The PR-AUC was above the baseline of 0.08 for other time-frames as well and indicates that performance better than random guessing is possible for this prediction task. Based on the confusion metrics we find that the models for all prediction time-frames are not suitable for practical purposes yet and more research is needed for further improvements.

6.2 Contribution

As far as we are aware, we are the first to investigate using clinical text in Dutch for prediction of involuntary admissions in patients suffering from severe psychiatric disorders. In this regard, we have explored NLP techniques along with traditional machine learning algorithms as well as the state-of-the-art deep transfer learning approaches. We also proposed a novel Siamese fine-tuning approach for training document feature extractors.

6.3 Limitations

We did not perform many experiments in order to select the best hyper-parameters for the deep learning models, and used the best hyper-parameters that were recommended in literature. We note that our work was severely limited by the computational resources and hence we were unable to fully exploit the power of deep transfer learning. For the Siamese fine-tuning, the training data-set was too small. While it was theoretically possible to create a large number of training examples, it was impractical for us to do so. As a result, we believe, we were not able to effectively evaluate and demonstrate the full potential of Siamese fine-tuning.

6.4 Future Work

In this work we observed that patient history corresponding to the last month of the prediction time-frame was more useful in predicting involuntary admissions. This finding could be explored further to determine the most effective amount of patient history that should be considered for this prediction task.

The full potential of the proposed fine-tuning approach could not be evaluated in this study. In the future, the approach could be evaluated in settings without any operational constraints. Currently we used the binary cross entropy loss to fine-tune the Siamese model. Future work could explore using triplet loss to fine-tune the model. While we did use a BERT model trained on multi-genre data to account for different writing styles, clinical data could be specifically used to further pre-train the model. Another related direction for future work could be incorporating more domain specific vocabulary while fine-tuning the model. In addition, other more representative clinical data-sets could be used to validate the findings of this study.

References

Abbe, A., Grouin, C., Zweigenbaum, P., and Falissard, B. (2015). Text mining applications in psychiatry: A systematic literature review. *International journal of methods in psychiatric research*, 25. 6

Aggarwal, C. and Zhai, C. (2012). *Mining Text Data*. 6, 7, 20

Ashwinkumar, U. and Anandakumar, K. (2010). Ethical and legal issues for medical data mining. *International Journal of Computer Applications*, 1. 12

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32. 21

Conner, K., Bohnert, A., McCarthy, J., Valenstein, M., Bossarte, R., Ignacio, R., Lu, N., and Ilgen, M. (2012). Mental disorder comorbidity and suicide among 2.96 million men receiving care in the veterans health administration health system. *Journal of abnormal psychology*, 122. 5

Cook, B., Progovac, A., Chen, P., Mullin, B., Hou, S., and Baca-Garcia, E. (2016). Novel use of natural language processing (nlp) to predict suicidal ideation and psychiatric symptoms in a text-based mental health intervention in madrid. *Computational and Mathematical Methods in Medicine*, 2016. 5

Demner-Fushman, D., Chapman, W. W., and McDonald, C. J. (2009). What can natural language processing do for clinical decision support? *Journal of Biomedical Informatics*, 42(5):760 – 772. Biomedical Natural Language Processing. 5

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805. 7

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655. 7

Eisenschlos, J., Ruder, S., Czapla, P., Kadras, M., Gugger, S., and Howard, J. (2019). Multifit: Efficient multi-lingual language model fine-tuning. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 7

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211. 6

Eriksson, R., Jensen, P. B., Frankild, S., Jensen, L. J., and Brunak, S. (2013). Dictionary construction and identification of possible adverse drug events in danish clinical narrative text. *Journal of the American Medical Informatics Association*, 20(5):947–953. 6

Ford, E., Carroll, J. A., Smith, H. E., Scott, D., and Cassell, J. A. (2016). Extracting information from the text of electronic medical records to improve case detection: a systematic review. *Journal of the American Medical Informatics Association*, 23(5):1007–1015. 6

Friedman, C., Rindflesch, T. C., and Corn, M. (2013). Natural language processing: State of the art and prospects for significant progress, a workshop sponsored by the national library of medicine. *Journal of Biomedical Informatics*, 46(5):765 – 773. 3

Friedman, S., Margolis, R., David, O., and Kesselman, M. (1983). Predicting psychiatric admission from an emergency room. psychiatric, psychosocial, and methodological factors. *The Journal of nervous and mental disease*, 171(3):155—158. 5

Goldberg, Y. (2016). A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420. 6

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://

www.deeplearningbook.org. 21, 23, 24, 29

Hao, X., Zhang, G., and Ma, S. (2016). Deep learning. *Int. J. Semantic Computing*, 10:417–. 6

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In *ACL*. 7

Huang, S. H., LePendu, P., Iyer, S. V., Tai-Seale, M., Carrell, D., and Shah, N. H. (2014). Toward personalizing treatment for depression: predicting diagnosis and severity. *Journal of the American Medical Informatics Association*, 21(6):1069–1075. 5

Jensen, P. B., Jensen, L. J., and Brunak, S. (2012). Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 13(6):395–405. 1

Jungfer, H.-A., Schneeberger, A. R., Borgwardt, S., Walter, M., Vogel, M., Gairing, S. K., Lang, U. E., and Huber, C. G. (2014). Reduction of seclusion on a hospital-wide level: successful implementation of a less restrictive policy. *Journal of psychiatric research*, 54:94–99. 1

Jurafsky, D. (2000). *Speech & language processing*. Pearson Education India. 20

Karystianis, G., Nevado, A., Kim, C.-H., Dehghan, A., Keane, J., and Nenadic, G. (2017). Automatic mining of symptom severity from psychiatric evaluation notes. *International journal of methods in psychiatric research*, 27. 1

Kaymak, U., Ben-David, A., and Potharst, R. (2010). Auk: a simple alternative to the auc. *Erasmus Research Institute of Management (ERIM), ERIM is the joint research institute of the Rotterdam School of Management, Erasmus University and the Erasmus School of Economics (ESE) at Erasmus Uni, Research Paper*, 25. 30

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. 29

Koch, G. R. (2015). Siamese neural networks for one-shot image recognition. 8

LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995. 6

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444. 21

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440. 7

Lyons, J. S., Stutesman, J., Neme, J., Vessey, J. T., O'Mahoney, M. T., and Camper, H. J. (1997). Predicting psychiatric emergency admissions and hospital outcome. *Medical Care*, 35(8):792–800. 5

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK. 6, 14, 16, 17

Melton, G. B. and Hripcsak, G. (2005). Automated detection of adverse events using natural language processing of discharge summaries. *Journal of the American Medical Informatics Association*, 12(4):448–457. 5

Menger, V., F.E., S., van Wijk L.M., and M., S. (2018a). DEDUCE: A pattern matching method for automatic de-identification of Dutch medical text. *Telematics and Informatics*, 35(4):727 – 736. 11

Menger, V., Scheepers, F., and Spruit, M. (2018b). Comparing deep learning and classical machine learning approaches for predicting inpatient violence incidents from clinical text. *Applied Sciences*, 8:981. 5

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119. 7

Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., and Jin, Z. (2016). How transferable are neural networks in nlp applications? *arXiv preprint arXiv:1603.06111*. 7

Mulder, C., Broer, J., Uitenbroek, D., Marle, P., van Hemert, A., and Wierdsma, A. I. (2006). [accelerated increase in the number of involuntary admissions following the implementation of the dutch act on compulsory admission to psychiatric hospitals (bopz)]. *Nederlands tijdschrift voor geneeskunde*, 150:319–22. 2

Murff, H. J., FitzHenry, F., Matheny, M. E., Gentry, N., Kotter, K. L., Crimin, K., Dittus,

R. S., Rosen, A. K., Elkin, P. L., Brown, S. H., et al. (2011). Automated identification of postoperative complications within an electronic medical record using natural language processing. *Jama*, 306(8):848–855. 5

Olfson, M., Ascher-Svanum, H., Faries, D., and Marcus, S. (2011). Predicting psychiatric hospital admission among adults with schizophrenia. *Psychiatric services (Washington, D.C.)*, 62:1138–45. 5

Pereira, L., Rijo, R., Silva, C., and Martinho, R. (2015). Text mining applied to electronic medical records: A literature review. *International Journal of E-Health and Medical Communications (IJEHMC)*, 6:1–18. 3

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proc. of NAACL*. 8

Poulin, C., Shiner, B., Thompson, P., Vepstas, L., Young-Xu, Y., Goertzel, B., Watts, B., Flashman, L., and Mcallister, T. (2014). Predicting the risk of suicide by analyzing the text of clinical notes. *PloS one*, 9:e85733. 5

Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: an astounding baseline for recognition. *Arxiv*. 7

Shickel, B., Tighe, P., Bihorac, A., and Rashidi, P. (2017). Deep ehr: A survey of recent advances on deep learning techniques for electronic health record (ehr) analysis. *Journal of Biomedical and Health Informatics.*, PP. 6

Steinert, T., Noorthoorn, E. O., and Mulder, C. L. (2014). The use of coercive interventions in mental health care in germany and the netherlands. a comparison of the developments in two neighboring countries. *Frontiers in Public Health*, 2:141. 1

Sun, A., Lim, E.-P., and Liu, Y. (2009). On strategies for imbalanced text classification using svm: A comparative study. *Decision Support Systems*, 48:191–201. 6, 7

Sun, C., Qiu, X., Xu, Y., and Huang, X. (2019). How to fine-tune bert for text classification? *Chinese Computational Linguistics*, page 194–206. 8, 24, 29

Vries, W., Cranenburgh, A., Bisazza, A., Caselli, T., van Noord, G., and Nissim, M. (2019). Bertje: A dutch bert model. 23

Wade, D. (2007). Ethics of collecting and using healthcare data. *BMJ*, 334(7608):1330–1331. 12

Wang, Y., Wang, L., Rastegar-Mojarad, M., Moon, S., Shen, F., Afzal, N., Liu, S., Zeng, Y., Mehrabi, S., Sohn, S., and Liu, H. (2018). Clinical information extraction applications: A literature review. *Journal of Biomedical Informatics*, 77:34 – 49. 3

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*. 23

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328. 7

Appendix A

A List of Dutch Stopwords with English Translations

Table A.1: A list of Dutch stopwords and their English meanings

Word	Meaning in English	Word	Meaning in English
de	the	deze	these
en	and	u	you
van	from, of, by	want	because
ik	I	nog	yet, still
te	to	zal	shall
dat	that	me	me, I
die	that, which, who	zij	she, they, it
in	in, into, at	nu	now
een	a, an	ge	-
hij	he	geen	no, none, neither
het	the	iets	something, anything, somewhat
niet	no, not	worden	be, become
zijn	to be, his, its, her	toch	yet, still, however
is	be	al	already, all, every
was	was, wash	waren	goods
op	on, up	veel	many
aan	to, on, at	meer	more
met	with, by, on	doen	to do, make
als	as, if, when	toen	then, when, as
voor	for, before	moet	must
had	had	ben	am
er	there	zonder	without
maar	but, only	kan	can
om	to, for, at	hun	their, them
hem	him	dus	so, therefore, thus
dan	than	alles	all, everything
zou	will, shall	onder	under, amongst
of	or, either, whether	ja	yes
wat	what, which, some	eens	once
mijn	my, mine	hier	here
dit	this, it	wie	who, which
zo	so, that, thus	werd	became

Table continues below

Word	Meaning in English	Word	Meaning in English
door	by, through, from	altijd	always
over	about, on, over	doch	but, yet
ze	she, they, them	wordt	is becoming
zich	herself, himself, itself, themselves	wezen	being
bij	at, in, to, bee	kunnen	can, may, able to
ook	also	onze	our
tot	until, to, for	zelf	self
je	you, your	tegen	at, against
mij	my, I	na	after, on, behind
uit	from, out, in	reeds	already
der	-	wil	want
daar	there	kon	could
haar	her, its, their	niets	nil, nothing
naar	to, for, at	uw	your
heb	have	iemand	someone
hoe	how	geweest	been
heeft	has	andere	others, another, else
hebben	have	omdat	because
stopwords from nltk.stopwords('dutch')			

Appendix B

Precision Recall Plots

B.1 Precision-Recall Plots for Classical ML Classifiers

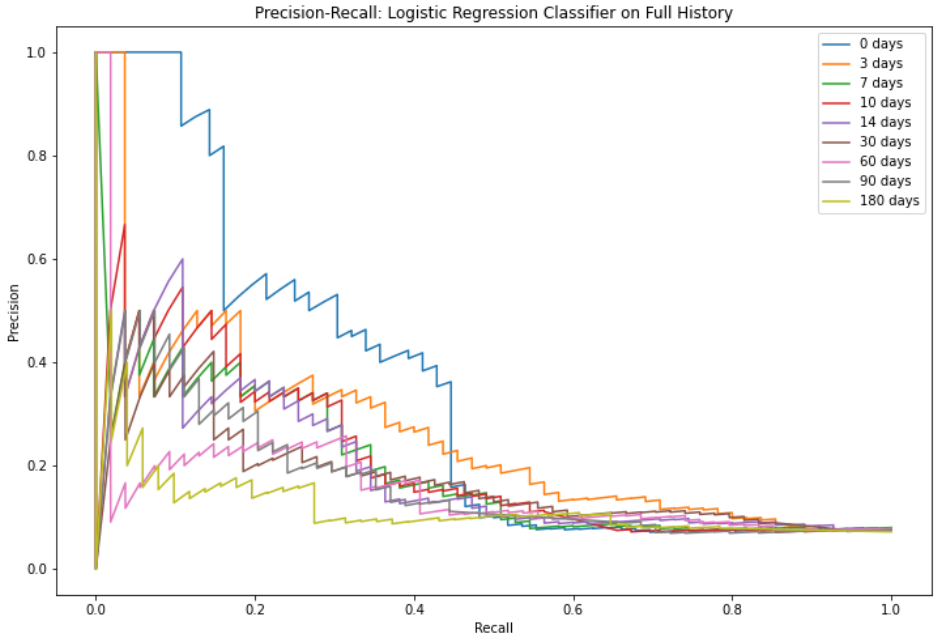

Figure B.1: Precision-Recall Curves for Logistic Regression Classifiers (Full History)

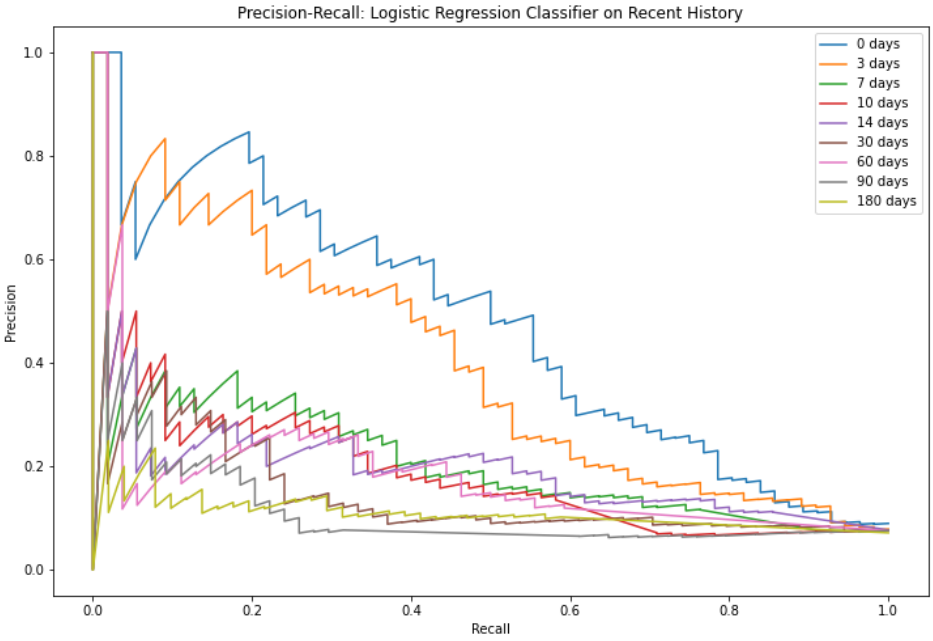

Figure B.2: Precision-Recall Curves for Logistic Regression Classifiers (Last Month History)

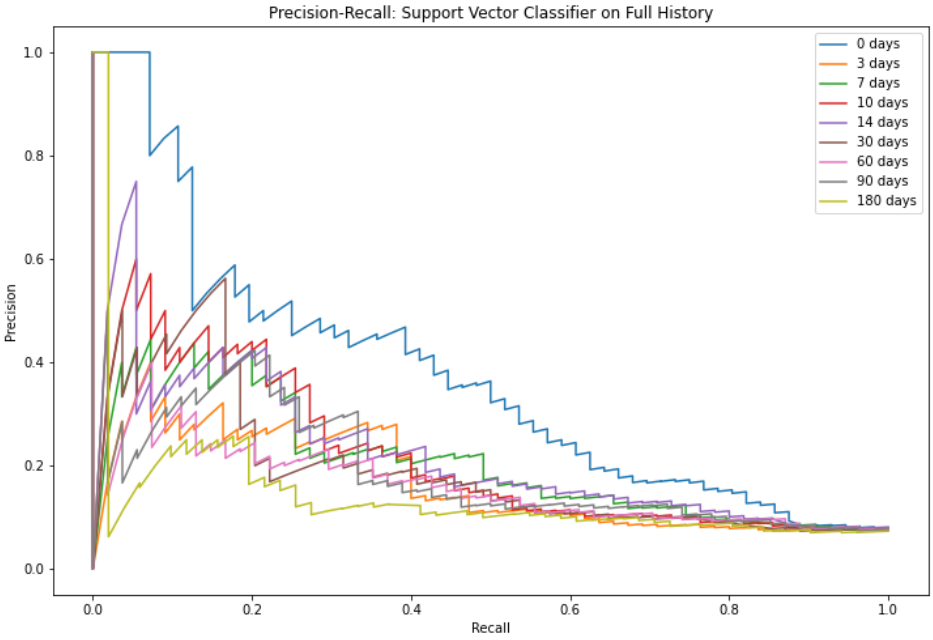

Figure B.3: Precision-Recall Curves for Support Vector Classifier (Full History)

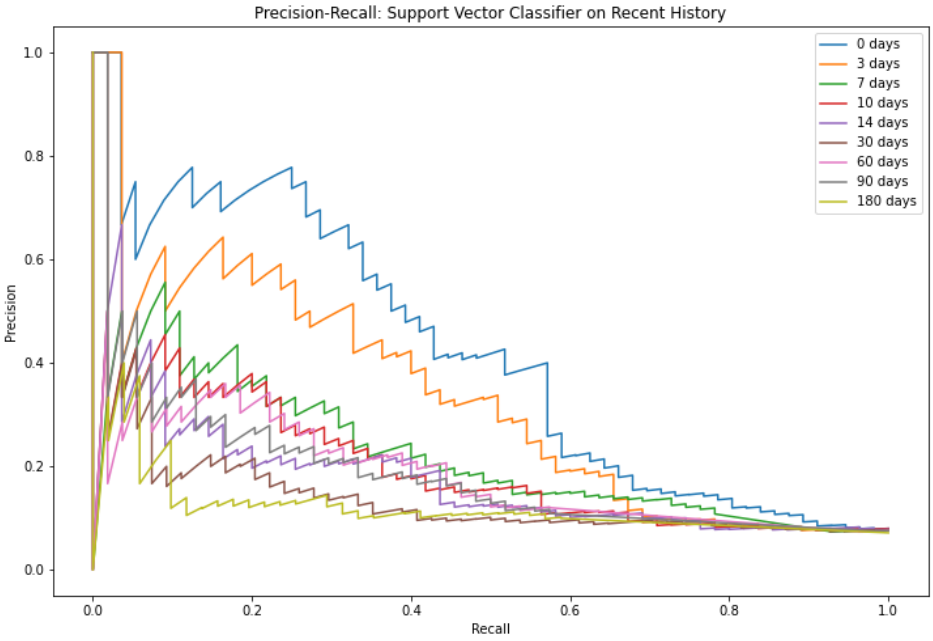

Figure B.4: Precision-Recall Curves for Support Vector Classifier (Last Month History)

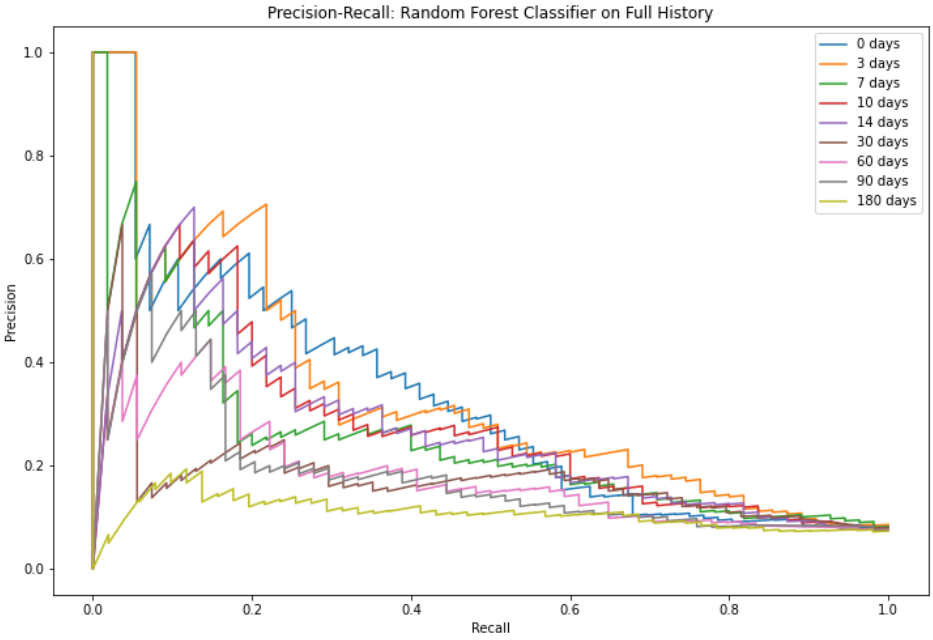

Figure B.5: Precision-Recall Curves for Random Forest Classifier (Full History)

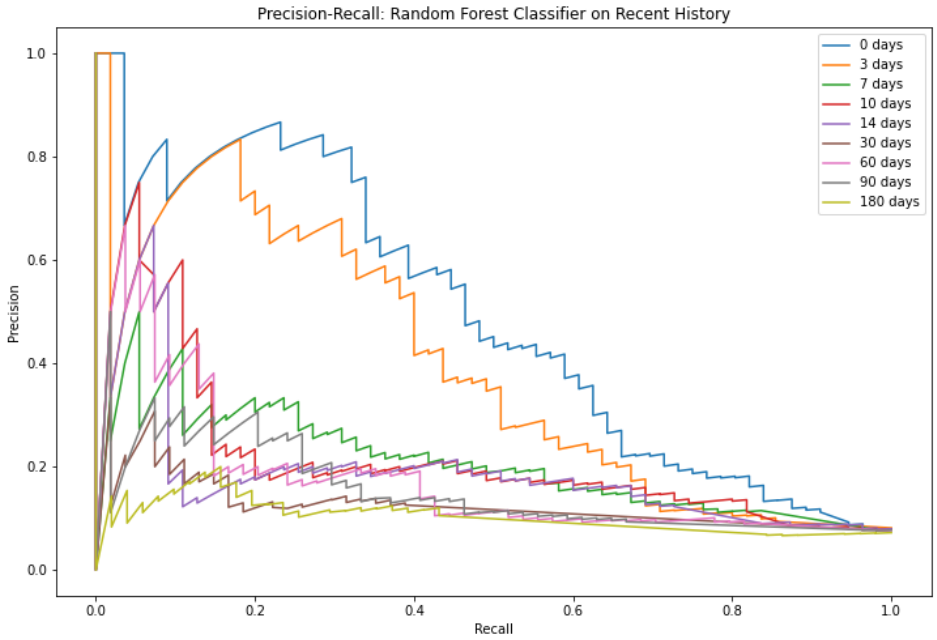

Figure B.6: Precision-Recall Curves for Random Forest Classifier (Last Month History)

B.2 Precision-Recall Plots for BERTje as a Classifier

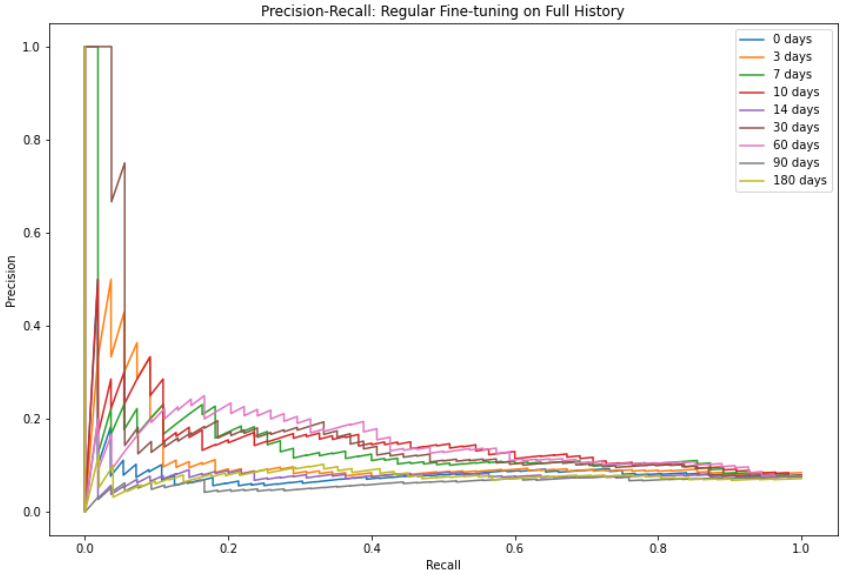

Figure B.7: Precision-Recall Curves: BERTje as a Classifier (Full History)

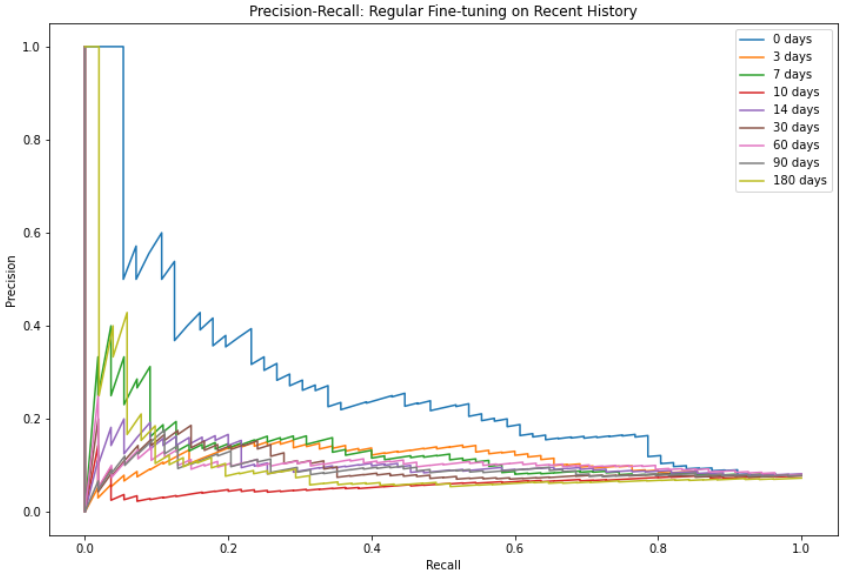

Figure B.8: Precision-Recall Curves: BERTje as a Classifier (Last Month History)

B.3 Precision-Recall Plots for BERTje as a feature extractor followed by Classical ML Classifiers

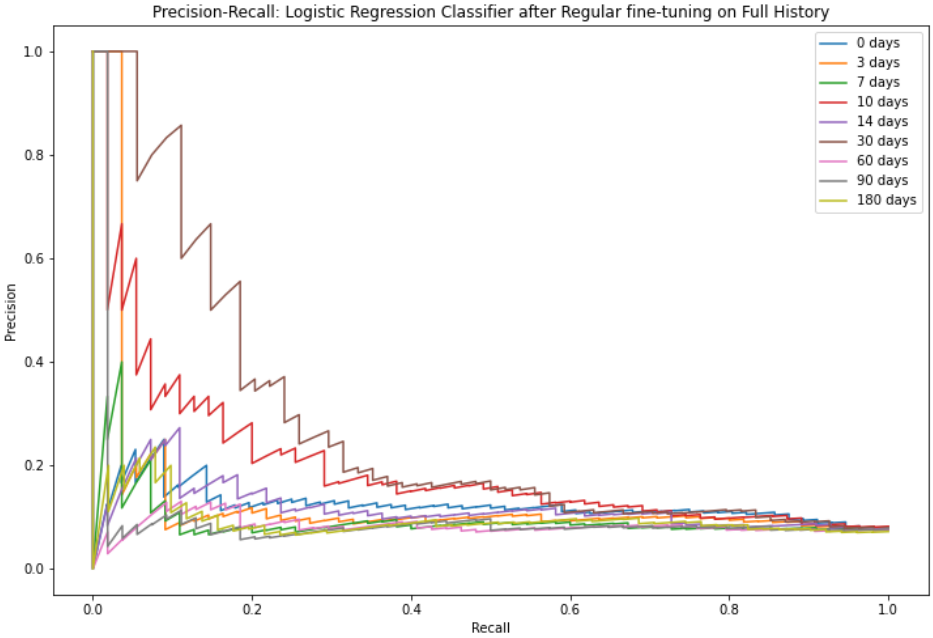

Figure B.9: Precision-Recall Curves for Logistic Regression Classifier on Embeddings after Regular Fine-tuning (Full History)

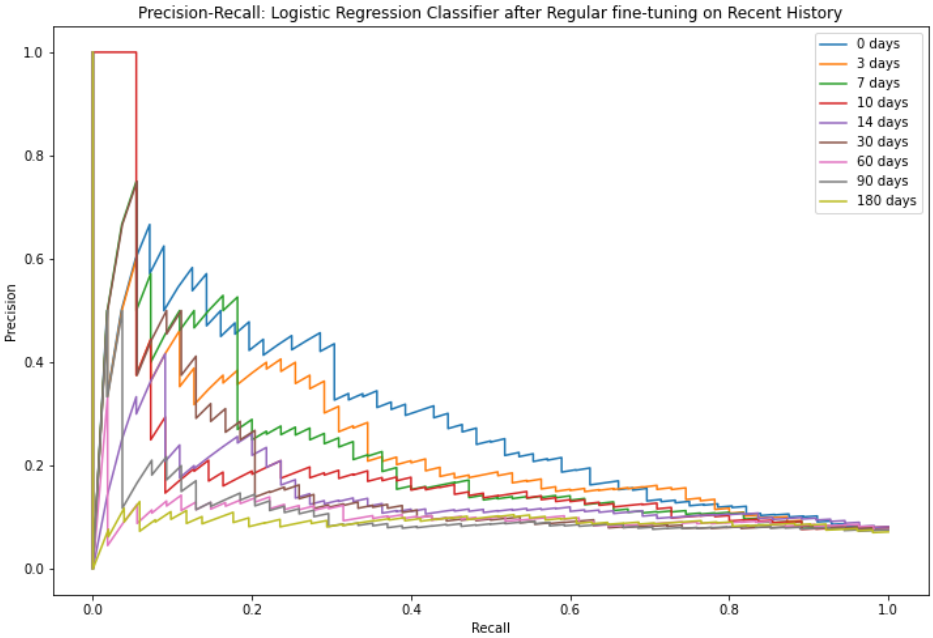

Figure B.10: Precision-Recall Curves for Logistic Regression Classifier on Embeddings after Regular Fine-tuning (Last Month History)

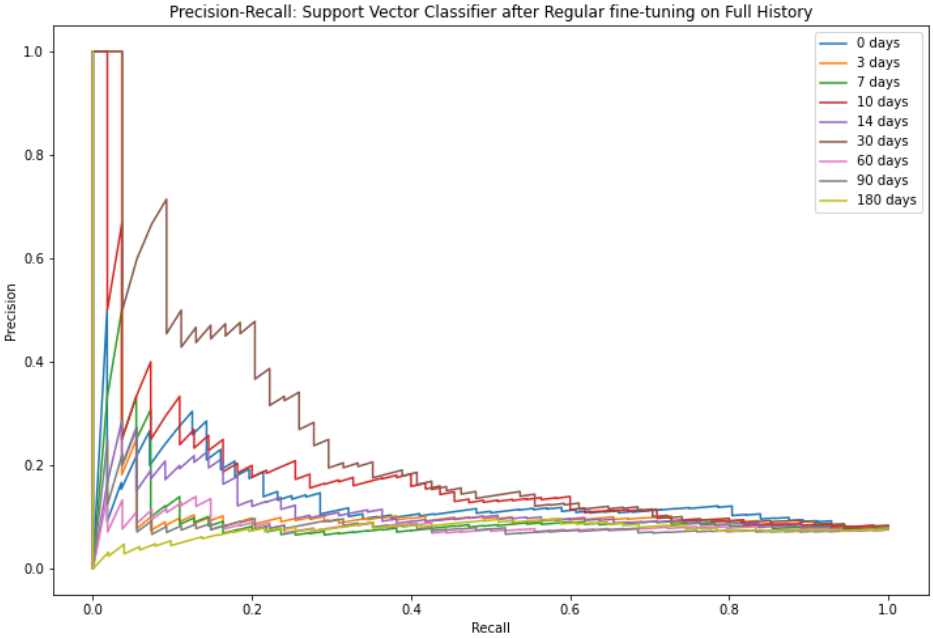

Figure B.11: Precision-Recall Curves for Support Vector Classifier on Embeddings after Regular Fine-tuning (Full History)

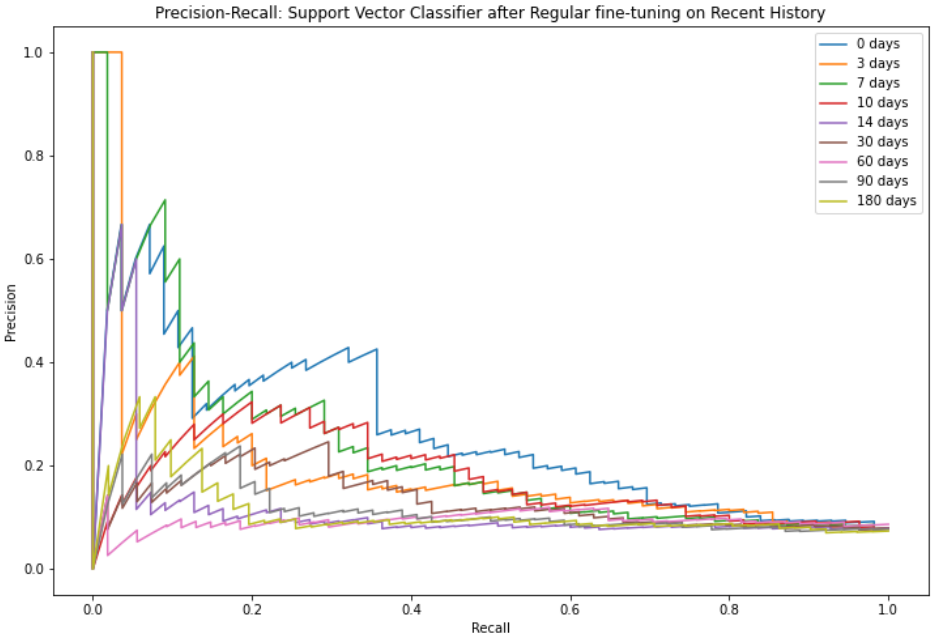

Figure B.12: Precision-Recall Curves for Support Vector Classifier on Embeddings after Regular Fine-tuning (Last Month History)

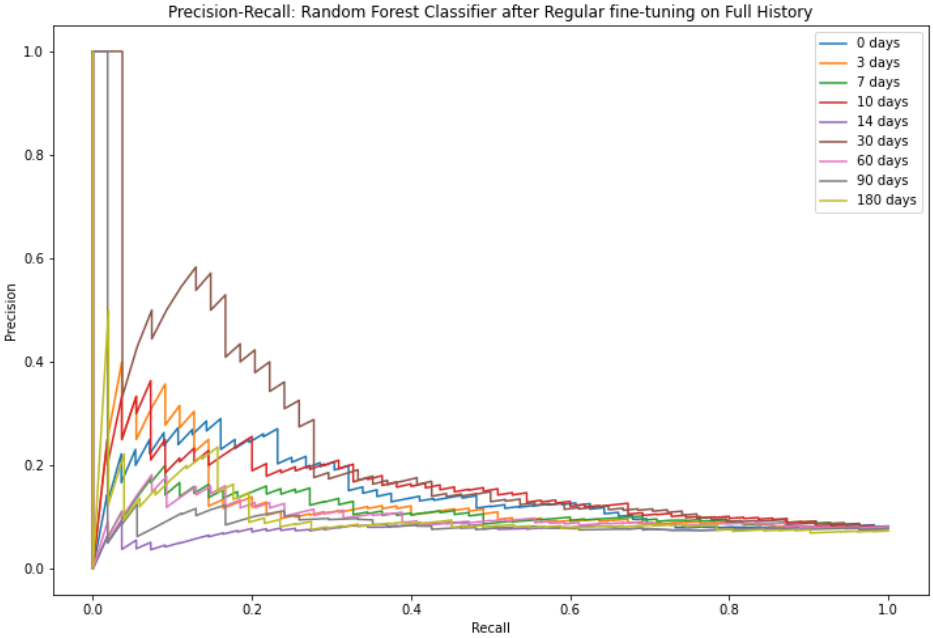

Figure B.13: Precision-Recall Curves for Random Forest Classifier on Embeddings after Regular Fine-tuning (Full History)

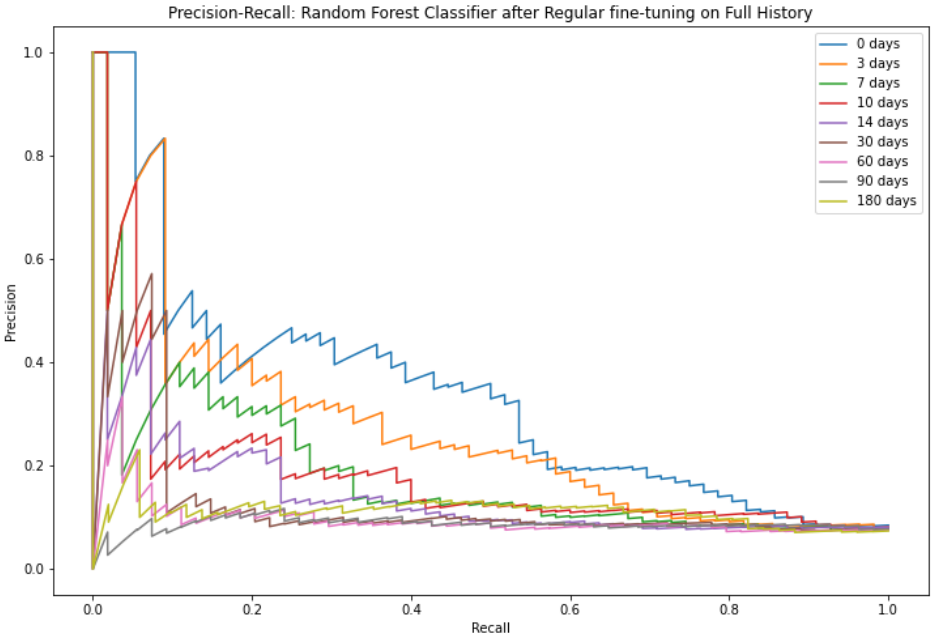

Figure B.14: Precision-Recall Curves for Random Forest Classifier on Embeddings after Regular Fine-tuning (Last Month History)

B.4 Precision-Recall Plots for BERTje as a feature extractor after Siamese Fine-tuning, followed by Classical ML Classifiers

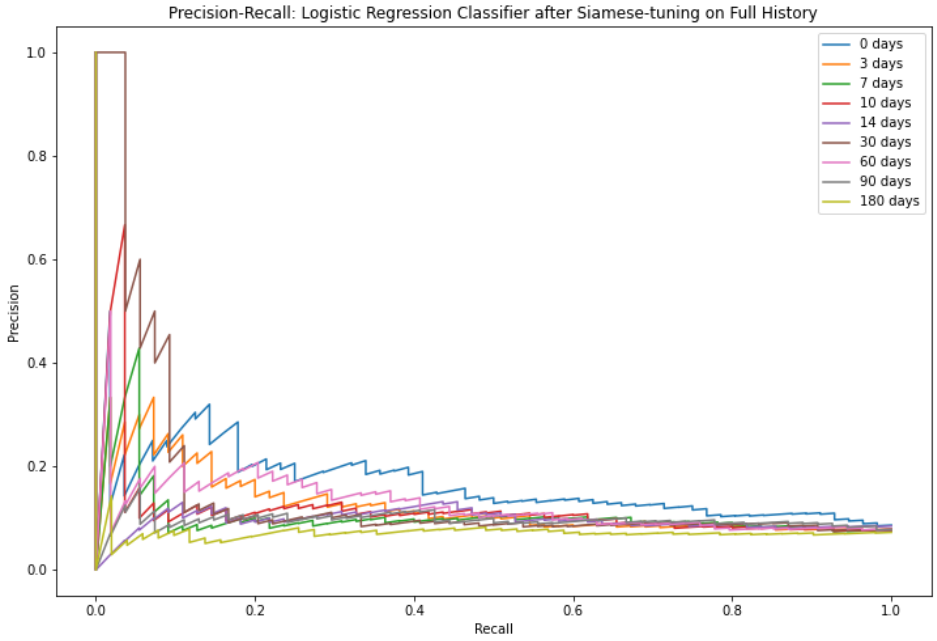

Figure B.15: Precision-Recall Curves for Logistic Regression Classifier on Embeddings after Siamese Fine-tuning (Full History)

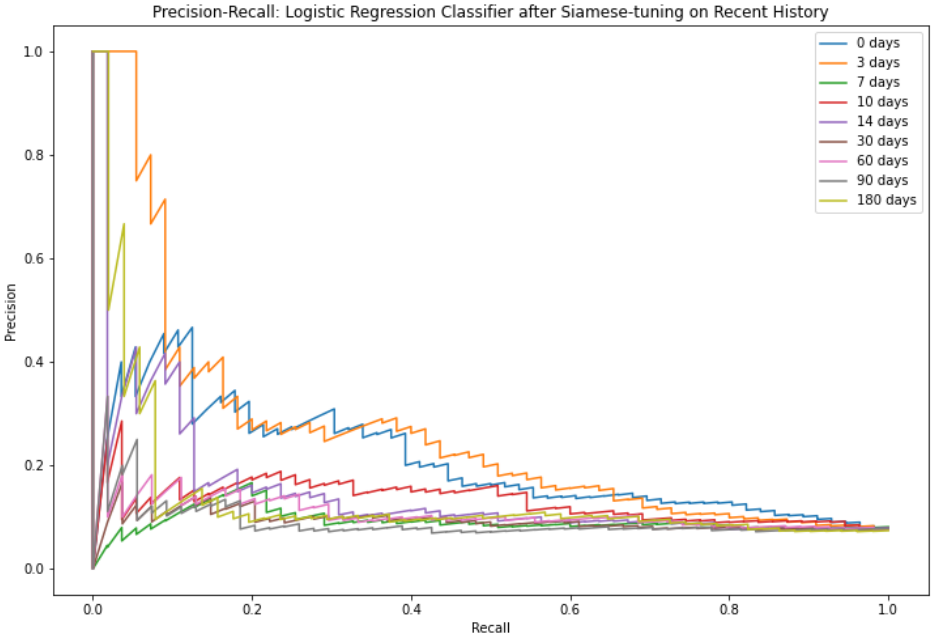

Figure B.16: Precision-Recall Curves for Logistic Regression Classifier on Embeddings after Siamese Fine-tuning (Last Month History)

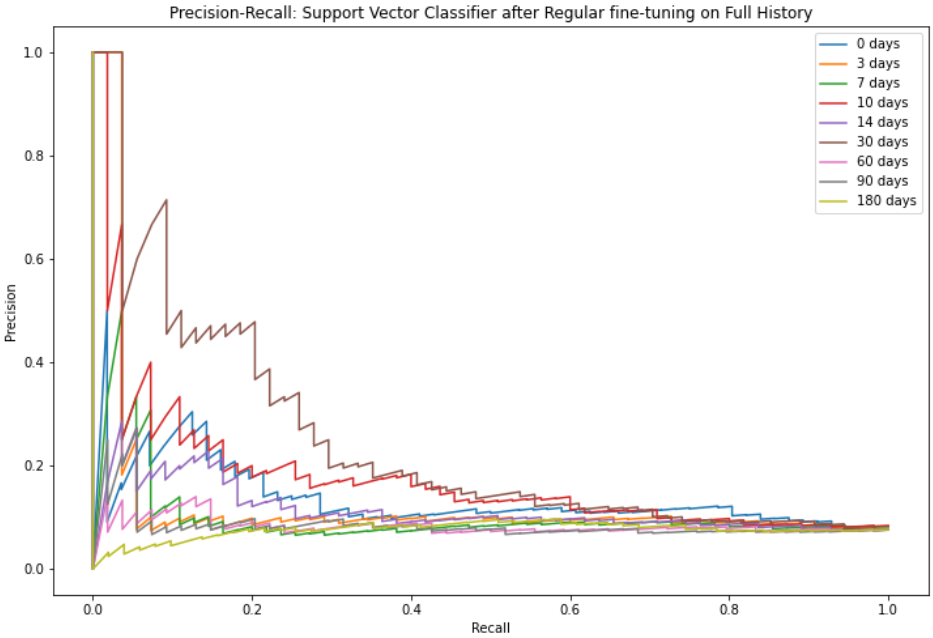

Figure B.17: Precision-Recall Curves for Support Vector Classifier on Embeddings after Siamese Fine-tuning (Full History)

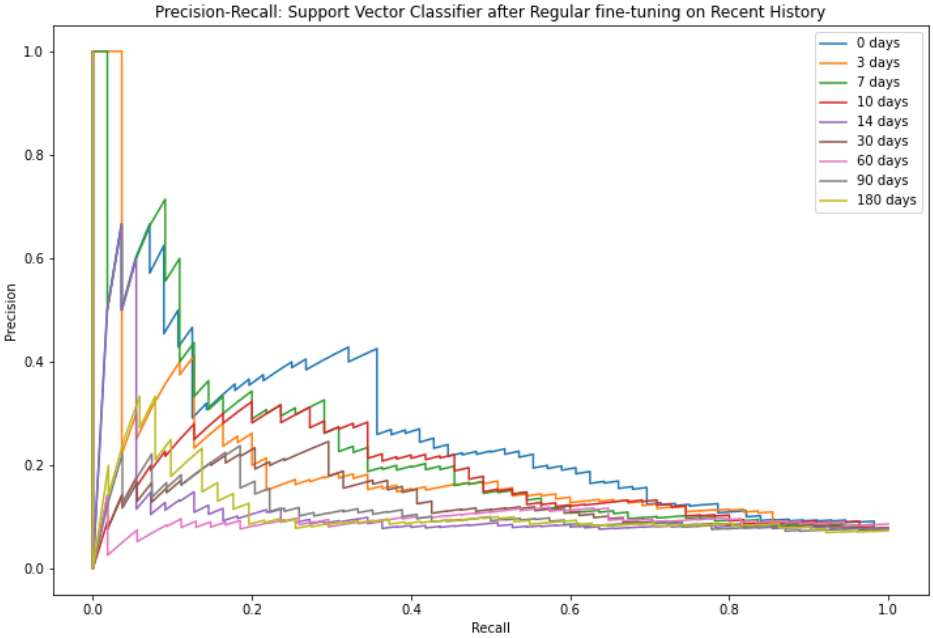

Figure B.18: Precision-Recall Curves for Support Vector Classifier on Embeddings after Siamese Fine-tuning (Last Month History)

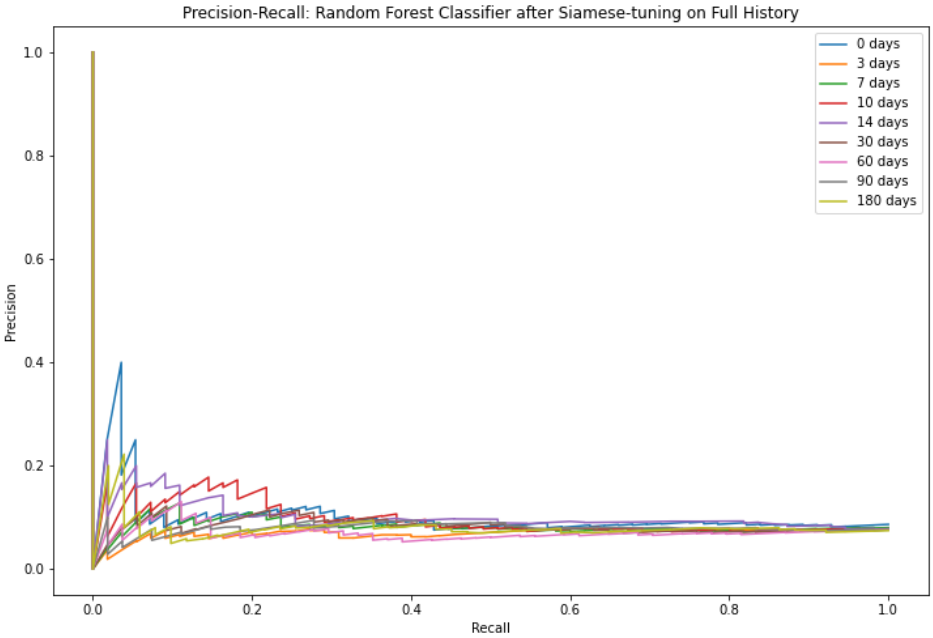

Figure B.19: Precision-Recall Curves for Random Forest Classifier on Embeddings after Siamese Fine-tuning (Full History)

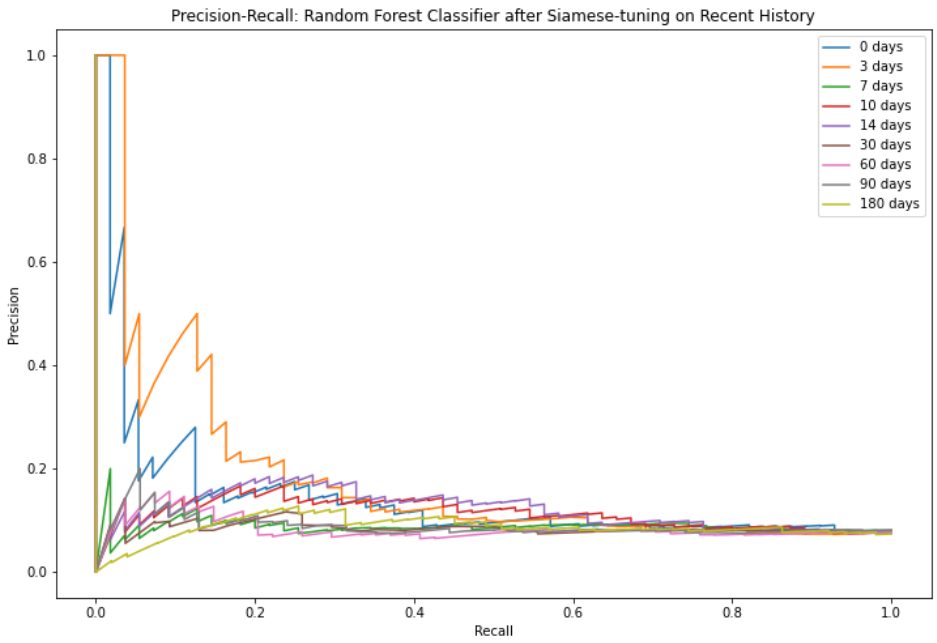

Figure B.20: Precision-Recall Curves for Random Forest Classifier on Embeddings after Siamese Fine-tuning (Last Month History)

Appendix C

Results: Classical Machine Learning Models

Here we provide the model metrics for the logistic regression, support vector machine, and random forest classifiers under various input representations. The metrics are provided per dataset for BOW and Tf-idf representations. Further, the metrics using full and the most recent past month patient history that is available for the respective time-periods is also given. In general, model performance is better when using data from the past month, and the better models tend to prefer a lower number of words as input features. Model performance also declines as the prediction time-frames move further away from the time of involuntary admission. Note that in the tables that follow, 'recent' is used to denote 'the last month data of the respective time-period'.

C.1 Models for all data-sets using different feature representations

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.46	0.5	0.28	0.11	0.07	0.09	0.17	0.12	0.14
2500-Recent History	0.65	0.93	0.62	0.27	0.23	0.09	0.38	0.37	0.16
1500-Full History	0.28	0.56	0.21	0.2	0.09	0.14	0.23	0.15	0.17
1500-Recent History	0.71	0.80	0.58	0.3	0.14	0.2	0.42	0.24	0.29
1000-Full History	0.45	0.55	0.10	0.18	0.11	0.07	0.26	0.18	0.08
1000-Recent History	0.58	0.83	0.48	0.32	0.18	0.18	0.41	0.29	0.26
750-Full History	0.39	0.58	0.19	0.16	0.12	0.14	0.23	0.21	0.16
750-Recent History	0.55	0.85	0.34	0.38	0.20	0.18	0.45	0.32	0.24
500-Full History	0.42	0.7	0.27	0.2	0.12	0.29	0.27	0.21	0.28
500-Recent History	0.57	0.86	0.36	0.38	0.21	0.29	0.45	0.34	0.32
250-Full History	0.54	0.67	0.37	0.25	0.14	0.18	**0.34**	0.24	0.24
250-Recent History	0.58	0.87	0.45	0.38	0.23	0.43	0.46	0.37	0.44
100-Full History	0.28	0,58	0.29	0.39	0.2	0.39	0.33	**0.29**	0.33
100-Recent History	0.59	0.81	0.50	0.41	0.23	0.39	**0.48**	0.36	**0.44**

Table C.1: Classifier Scores for Dataset: 0 days before admission with BOW representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.33	1.00	0.44	0.18	0.05	0.21	0.24	0.10	0.29
2500-Recent History	0.53	0.89	0.57	0.34	0.29	0.14	0.41	0.43	0.23
1500-Full History	0.23	1.00	0.20	0.12	0.04	0.16	0.16	0.07	0.18
1500-Recent History	0.58	0.83	0.42	0.34	0.27	0.20	0.43	0.29	0.21
1000-Full History	0.23	1.00	0.20	0.14	0.09	0.21	0.18	0.16	0.21
1000-Recent History	0.51	0.79	0.55	0.34	0.27	0.20	0.41	0.40	0.29
750-Full History	0.29	1.00	0.29	0.14	0.11	0.20	0.19	0.19	0.23
750-Recent History	0.69	0.87	0.43	0.32	0.23	0.16	0.44	0.37	0.23
500-Recent History	0.52	0.67	0.30	0.20	0.11	0.14	0.29	0.18	0.19
500-Recent History	0.47	0.88	0.41	0.39	0.25	0.29	0.43	0.39	0.34
250-Full History	0.42	0.88	0.33	0.29	0.12	0.16	0.34	0.22	0.22
250-Recent History	0.51	0.77	0.38	0.39	0.30	0.34	0.44	0.44	0.36
100-Full History	0.48	0.58	0.46	0.21	0.12	0.29	0.30	0.21	**0.35**
100-Recent History	0.56	0.82	0.48	0.36	0.32	0.39	0.43	**0.46**	0.43

Table C.2: Classifier Scores for Data-set: 0 days before admission with Tf-idf representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.27	1.00	0.26	0.2	0.04	0.22	0.23	0.07	0.24
2500-Recent History	0.36	0.88	0.2	0.22	0.13	0.02	0.37	0.22	0.03
1500-Full History	0.24	0.00	0.22	0.15	0.00	0.15	0.18	0.00	0.17
1500-Recent History	0.43	0.90	0.36	0.22	0.16	0.07	0.29	0.28	0.12
1000-Full History	0.26	0.00	0.17	0.18	0.00	0.13	0.21	0.00	0.15
1000-Recent History	0.45	0.75	0.36	0.18	0.11	0.15	0.26	0.19	0.21
750-Full History	0.42	0.00	0.31	0.20	0.00	0.22	0.27	0.00	0.26
750-Recent History	0.46	0.67	0.27	0.20	0.07	0.18	0.28	0.13	0.22
500-Full History	0.17	1.00	0.16	0.27	0.05	0.22	0.21	0.10	0.18
500-Recent History	0.44	0.78	0.20	0.31	0.13	0.18	0.36	0.22	0.19
250-Full History	0.32	0.83	0.26	0.31	0.09	0.38	0.31	0.16	0.31
250-Recent History	0.50	0.71	0.42	0.40	0.18	0.36	**0.44**	0.29	**0.39**
100-Full History	0.42	0.90	0.26	0.20	0.16	0.38	0.27	0.28	**0.31**
100-Recent History	0.45	0.75	0.35	0.40	0.27	0.38	0.42	0.40	0.37

Table C.3: Classifier Scores for Dataset: 3 days before admission with BOW representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.33	0.00	0.19	0.16	0.00	0.09	0.22	0.00	0.12
2500-Recent History	0.39	0.92	0.20	0.16	0.22	0.02	0.23	0.35	0.03
1500-Full History	0.18	0.00	0.20	0.11	0.00	0.11	0.14	0.00	0.14
1500-Recent History	0.38	0.85	0.50	0.18	0.20	0.11	0.25	0.32	0.18
1000-Full History	0.22	1.00	0.22	0.15	0.02	0.15	0.17	0.04	0.18
1000-Recent History	0.31	0.82	0.44	0.18	0.27	0.15	0.23	0.27	0.22
750-Full History	0.37	1.00	0.37	0.13	0.02	0.24	0.19	0.04	0.29
750-Recent History	0.25	0.82	0.52	0.18	0.16	0.20	0.21	0.27	0.29
500-Full History	0.44	0.67	0.43	0.20	0.18	0.16	0.28	**0.29**	0.24
500-Recent History	0.33	0.71	0.40	0.25	0.22	0.18	0.29	0.33	0.25
250-Full History	0.50	0.88	0.30	0.24	0.13	0.13	0.32	0.22	0.18
250-Recent History	0.27	0.67	0.35	0.47	0.33	0.31	0.34	0.44	0.33
100-Full History	0.32	0.52	0.33	0.33	0.20	0.27	**0.32**	0.29	0.30
100-Recent History	0.33	0.69	0.32	0.36	0.33	0.35	0.34	**0.44**	0.33

Table C.4: Classifier Scores for Dataset: 3 days before admission with Tf-idf representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.26	0.00	0.24	0.13	0.00	0.13	0.17	0.00	0.17
2500-Recent History	0.27	0.00	0.33	0.18	0.00	0.04	0.22	0.00	0.07
1500-Full History	0.21	0.00	0.17	0.13	0.00	0.13	0.16	0.00	0.15
1500-Recent History	0.37	0.50	0.38	0.13	0.02	0.15	0.19	0.04	0.21
1000-Full History	0.15	1.00	0.12	0.09	0.05	0.11	0.11	0.10	0.12
1000-Recent History	0.26	0.50	0.22	0.13	0.02	0.11	0.17	0.04	0.15
750-Full History	0.24	0.00	0.21	0.15	0.00	0.13	0.18	0.00	0.16
750-Recent History	0.33	0.00	0.18	0.11	0.00	0.09	0.16	0.00	0.12
500-Full History	0.19	0.00	0.14	0.18	0.00	0.22	0.18	0.00	0.17
500-Recent History	0.32	0.50	0.12	0.15	0.02	0.05	0.20	0.04	0.07
250-Full History	0.32	1.00	0.25	0.29	0.00	0.25	**0.30**	0.04	0.25
250-Recent History	0.29	0.67	0.26	0.22	0.04	0.18	0.25	0.07	0.21
100-Recent History	0.20	0.67	0.18	0.33	0.07	0.25	0.25	0.13	0.21
100-Recent History	0.38	0.43	0.30	0.18	0.05	0.16	0.25	0.10	0.21

Table C.5: Classifier Scores for Dataset: 7 days before admission with BOW representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.14	0.00	0.22	0.05	0.00	0.07	0.08	0.00	0.11
2500-Recent History	0.21	0.40	0.20	0.09	0.04	0.02	0.13	0.07	0.03
1500-Full History	0.08	0.00	0.22	0.07	0.00	0.15	0.08	0.00	0.18
1500-Recent History	0.12	0.67	0.27	0.05	0.04	0.05	0.08	0.07	0.09
1000-Full History	0.14	0.64	0.20	0.05	0.20	0.11	0.08	0.32	0.14
1000-Recent History	0.11	0.67	0.14	0.05	0.04	0.04	0.07	0.27	0.06
750-Full History	0.27	0.64	0.29	0.15	0.13	0.18	0.19	**0.21**	0.22
750-Recent History	0.15	0.50	0.21	0.16	0.04	0.07	0.16	0.07	0.11
500-Full History	0.21	0.58	0.25	0.07	0.13	0.11	0.11	0.21	0.15
500-Recent History	0.15	0.31	0.27	0.22	0,07	0.11	0.18	0.12	0.16
250-Full History	0.21	0.40	0.20	0.42	0.15	0.40	0.28	0.21	**0.27**
250-Recent History	0.18	0.38	0.28	0.27	0.09	0.20	0.22	0.15	0.23
100-Full History	0.31	0.5	0.27	0.22	0.04	0.16	0.26	0.07	0.20
100-Recent History	0.30	0.32	0.40	0.31	0.18	0.18	**0.31**	**0.23**	**0.25**

Table C.6: Classifier Scores for Data-set: 7 days before admission with Tf-idf representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.21	0.00	0.12	0.13	0.00	0.09	0.16	0.00	0.11
2500-Recent History	0.27	0.50	0.33	0.13	0.02	0.04	0.17	0.04	0.07
1500-Full History	0.19	0.00	0.21	0.13	0.00	0.16	0.15	0.00	0.19
1500-Recent History	0.24	0.75	0.42	0.13	0.05	0.09	0.17	0.10	0.15
1000-Full History	0.19	0.00	0.13	0.25	0.00	0.18	0.22	0.00	0.15
100-Recent History	0.14	0.00	0.09	0.05	0.00	0.04	0.08	0.00	0.05
750-Full History	0.13	0.00	0.12	0.25	0.00	0.16	0.17	0.00	0.14
750-Recent History	0.15	0.67	0.16	0.07	0.04	0.09	0.10	0.07	0.12
500-Full History	0.23	1.00	0.26	0.18	0.02	0.20	0.20	0.04	0.23
500-Recent History	0.29	0.50	0.19	0.11	0.02	0.13	0.16	0.04	0.15
250-Full History	0.38	1.00	0.22	0.22	0.02	0.36	0.28	0.04	**0.27**
250-Recent History	0.29	0.60	0.20	0.24	0.05	0.18	0.26	0.10	0.19
100-Full History	0.24	0.33	0.21	0.18	0.02	0.22	0.21	0.21	0.21
100-Recent History	0.32	0.50	0.35	0.18	0.07	0.22	0.23	0.13	**0.27**

Table C.7: Classifier Scores for Data-set: 10 days before admission with BOW representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.08	0.00	0.21	0.05	0.00	0.11	0.07	0.00	0.14
2500-Recent History	0.13	0.75	0.14	0.07	0.05	0.05	0.09	0.10	0.08
1500-Full History	0.09	0.00	0.14	0.04	0.00	0.09	0.05	0.00	0.11
1500-Recent History	0.15	0.60	0.15	0.09	0.05	0.04	0.11	0.10	0.06
1000-Full History	0.29	0.00	0.26	0.13	0.00	0.18	0.18	0.00	0.22
1000-Recent History	0.09	0.50	0.12	0.07	0.04	0.05	0.08	0.07	0.07
750-Full History	0.23	0.00	0.26	0.13	0.00	0.22	0.16	0.00	0.22
750-Recent History	0.12	0.00	0.09	0.13	0.00	0.05	0.12	0.00	0.07
500-Full History	0.23	0.53	0.35	0.22	0.15	0.20	0.22	0.23	0.26
500-Recent History	0.11	0.50	0.24	0.15	0.04	0.15	0.13	0.07	0.18
250-Full History	0.32	0.42	0.26	0.15	0.02	0.25	0.20	**0.27**	0.26
250-Recent History	0.16	0.56	0.17	0.20	0.09	0.09	0.18	**0.16**	0.12
100-Full History	0.33	0.28	0.30	0.25	0.24	0.18	**0.29**	0.03	0.23
100-Recent History	0.25	0.44	0.30	0.33	0.07	0.22	**0.28**	0.12	0.25

Table C.8: Classifier Scores for Data-set: 10 days before admission with tfidf representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.17	0.00	0.15	0.07	0.00	0.09	0.10	0.00	0.11
2500-Recent History	0.17	0.00	0.14	0.09	0.00	0.02	0.12	0.00	0.03
1500-Full History	0.21	0.00	0.19	0.09	0.00	0.11	0.13	0.00	0.14
1500-Recent History	0.14	1.00	0.23	0.09	0.02	0.05	0.11	0.04	0.09
1000-Full History	0.24	0.00	0.24	0.15	0.00	0.25	0.18	0.00	0.25
1000-Recent History	0.18	1.00	0.14	0.09	0.02	0.07	0.12	0.04	0.10
750-Full History	0.12	0.00	0.09	0.18	0.00	0.15	0.15	0.00	0.11
750-Recent History	0.19	1.00	0.18	0.07	0.02	0.13	0.11	0.04	0.15
500-Full History	0.09	0.00	0.16	0.16	0.00	0.13	0.11	0.00	0.14
500-Recent History	0.26	0.50	0.23	0.13	0.02	0.11	0.17	0.04	0.15
250-Full History	0.28	0.00	0.2	0.31	0.00	0.35	**0.29**	0.00	0.25
250-Recent History	0.24	0.50	0.17	0.11	0.02	0.13	0.15	0.04	0.14
100-Full History	0.25	0.00	0.24	0.20	0.24	0.29	0.22	0.00	0.26
100-Recent History	0.28	0.38	0.20	0.15	0.05	0.29	0.19	0.10	**0.24**

Table C.9: Classifier Scores for Data-set: 14 days before admission with BOW representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.07	0.00	0.07	0.04	0.00	0.04	0.05	0.00	0.05
2500-Recent History	0.19	0.50	0.33	0.09	0.04	0.11	0.12	0.07	0.16
1500-Full History	0.21	0.00	0.21	0.07	0.00	0.15	0.11	0.00	0.17
1500-Recent History	0.10	0.00	0.29	0.09	0.00	0.13	0.10	0.00	0.18
1000-Full History	0.35	0.83	0.28	0.20	0.09	0.20	0.26	0.16	0.23
1000-Recent History	0.09	0.00	0.20	0.09	0.00	0.05	0.09	0.00	0.09
750-Full History	0.26	0.60	0.21	0.16	0.11	0.20	0.20	0.18	0.21
750-Recent History	0.11	0.50	0.16	0.13	0.02	0.05	0.12	0.04	0.08
500-Full History	0.28	0.47	0.23	0.16	0.16	0.05	0.21	0.24	0.09
500-Recent History	0.10	0.50	0.22	0.13	0.02	0.13	0.11	0.04	0.16
250-Full History	0.30	0.41	0.30	0.25	0.22	0.25	0.27	**0.29**	**0.28**
250-Recent History	0.17	0.57	0.28	0.22	0.07	0.13	0.19	**0.13**	0.17
100-Full History	0.33	0.27	0.16	0.24	0.22	0.35	0.28	0.00	0.22
100-Recent History	0.25	0.22	0.19	0.22	0.04	0.24	**0.23**	0.06	0.21

Table C.10: Classifier Scores for Data-set: 14 days before admission with Tf-idf representation

Top-N words, Full/Recent Data	Precision			Recall			F1 score		
	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.22	0.00	0.19	0.13	0.00	0.13	0.16	0.00	0.15
2500-Recent History	0.24	0.00	0.20	0.13	0.00	0.02	**0.17**	0.00	0.03
1500-Full History	0.14	0.00	0.11	0.07	0.00	0.07	0.10	0.00	0.09
1500-Recent History	0.17	0.00	0.18	0.04	0.00	0.06	0.06	0.00	0.08
1000-Full History	0.21	0.00	0.15	0.15	0.00	0.17	0.17	0.00	0.16
1000-Recent History	0.12	0.00	0.16	0.06	0.00	0.06	0.08	0.00	0.08
750-Full History	0.12	0.00	0.08	0.09	0.00	0.09	0.11	0.00	0.08
750-Recent History	0.07	0.00	0.08	0.04	0.00	0.04	0.05	0.00	0.05
500-Full History	0.20	0.00	0.15	0.15	0.00	0.15	0.17	0.00	0.15
500-Recent History	0.14	0.67	0.16	0.06	0.04	0.06	0.08	0.07	0.08
250-Full History	0.14	0.00	0.19	0.44	0.00	0.30	0.21	0.00	0.23
250-Recent History	0.18	0.50	0.18	0.13	0.02	0.11	0.15	0.04	**0.14**
100-Full History	0.11	0.00	0.13	0.43	0.00	0.30	0.17	**0.24**	0.18
100-Recent History	0.18	0.67	0.16	0.09	0.07	0.11	0.12	0.13	0.13

Table C.11: Classifier Scores for Data-set: 30 days before admission with BOW representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.12	0.00	0.15	0.07	0.00	0.06	0.09	0.00	0.08
2500-Recent History	0.15	0.50	0.40	0.07	0.04	0.07	0.10	0.07	0.12
1500-Full History	0.11	0.67	0.13	0.06	0.07	0.07	0.07	0.13	0.10
1500-Recent History	0.07	0.50	0.20	0.07	0.04	0.04	0.07	0.07	0.06
1000-Full History	0.17	0.31	0.21	0.07	0.09	0.11	0.10	0.14	0.14
1000-Recent History	0.05	0.60	0.10	0.06	0.06	0.02	0.05	0.10	0.03
750-Full History	0.15	0.38	0.27	0.07	0.11	0.15	0.10	0.17	0.19
750-Recent History	0.05	0.60	0.00	0.06	0.06	0.00	0.06	0.10	0.00
500-Full History	0.18	0.41	0.15	0.07	0.13	0.15	0.11	0.20	0.05
500-Recent History	0.09	0.60	0.22	0.11	0.06	0.09	0.10	0.10	0.13
250-Full History	0.18	0.26	0.19	0.37	0.19	0.41	**0.24**	0.22	**0.26**
250-Recent History	0.13	0.25	0.15	0.17	0.09	0.09	0.15	0.14	0.11
100-Full History	0.28	0.16	0.19	0.19	0.17	0.33	0.21	0.00	0.24
100-Recent History	0.10	0.17	0.16	0.11	0.13	0.09	0.11	**0.15**	0.12

Table C.12: Classifier Scores for Data-set: 30 days before admission with Tf-idf representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.16	0.00	0.14	0.17	0.00	0.09	0.17	0.00	0.11
2500-Recent History	0.09	0.00	0.00	0.04	0.00	0.00	0.05	0.00	0.00
1500-Full History	0.09	0.00	0.08	0.06	0.00	0.06	0.07	0.00	0.07
1500-Recent History	0.11	0.00	0.17	0.06	0.00	0.02	0.07	0.00	0.03
1000-Full History	0.11	0.00	0.12	0.19	0.00	0.17	0.14	0.00	0.14
1000-Recent History	0.06	0.00	0.06	0.02	0.00	0.02	0.03	0.00	0.03
750-Full History	0.15	0.00	0.15	0.15	0.00	0.20	0.15	0.00	0.17
750-Recent History	0.04	0.75	0.13	0.02	0.06	0.06	0.03	0.10	0.08
500-Full History	0.24	0.00	0.22	0.15	0.00	0.24	0.18	0.00	0.23
500-Recent History	0.11	0.38	0.21	0.06	0.06	0.15	0.07	0.10	0.17
250-Full History	0.25	0.00	0.20	0.13	0.00	0.35	0.17	0.00	**0.26**
250-Recent History	0.14	0.00	0.14	0.11	0.00	0.13	0.12	0.00	0.14
100-Full History	0.24	0.00	0.22	0.19	0.00	0.30	**0.21**	0.17	0.25
100-Recent History	0.22	0.17	0.16	0.22	0.06	0.37	0.22	0.08	0.22

Table C.13: Classifier Scores for Data-set: 60 days before admission with BOW representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.19	0.00	0.10	0.06	0.00	0.06	0.09	0.00	0.07
2500-Recent History	0.05	0.00	0.00	0.04	0.00	0.00	0.04	0.00	0.00
1500-Full History	0.17	0.50	0.11	0.11	0.06	0.06	0.13	0.10	0.07
1500-Recent History	0.11	0.00	0.05	0.11	0.00	0.02	0.11	0.00	0.03
1000-Full History	0.27	0.62	0.28	0.11	0.09	0.20	0.16	0.16	0.24
1000-Recent History	0.06	0.33	0.06	0.07	0.09	0.02	0.07	0.14	0.03
750-Full History	0.31	0.44	0.24	0.07	0.07	0.17	0.12	0.13	0.20
750-Recent History	0.07	0.36	0.14	0.09	0.07	0.06	0.08	0.12	0.08
500-Full History	0.12	0.38	0.04	0.07	0.17	0.02	0.09	**0.23**	0.03
500-Recent History	0.10	0.33	0.19	0.13	0.13	0.09	0.11	**0.19**	0.12
250-Full History	0.26	0.24	0.17	0.17	0.17	0.35	0.20	0.20	0.23
250-Recent History	0.12	0.20	0.10	0.17	0.19	0.04	0.14	0.19	0.05
100-Full History	0.15	0.19	0.17	0.28	0.22	0.30	0.20	0.00	0.22
100-Recent History	0.21	0.08	0.21	0.35	0.22	0.39	**0.26**	0.12	**0.28**

Table C.14: Classifier Scores for Data-set: 60 days before admission with Tf-idf representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.04	0.00	0.08	0.02	0.00	0.06	0.03	0.00	0.07
2500-Recent History	0.19	0.00	0.00	0.09	0.00	0.00	0.12	0.00	0.00
1500-Full History	0.12	0.19	0.17	0.17	0.09	0.24	0.14	0.12	0.20
1500-Recent History	0.23	0.00	0.00	0.06	0.00	0.00	0.09	0.00	0.00
1000-Full History	0.11	0.00	0.10	0.20	0.00	0.26	0.14	0.00	0.15
1000-Recent History	0.12	0.33	0.18	0.02	0.02	0.04	0.03	0.04	0.06
750-Full History	0.23	0.00	0.21	0.15	0.00	0.22	0.18	0.00	0.22
750-Recent History	0.04	0.00	0.17	0.02	0.00	0.04	0.03	0.00	0.06
500-Full History	0.20	0.00	0.17	0.13	0.00	0.17	0.16	0.00	0.17
500-Recent History	0.13	0.00	0.14	0.06	0.00	0.06	0.08	0.00	0.08
250-Full History	0.32	0.26	0.34	0.15	0.11	0.22	0.20	0.16	**0.27**
250-Recent History	0.14	0.00	0.09	0.09	0.00	0.06	0.11	0.00	0.07
100-Full History	0.21	0.29	0.23	0.15	0.13	0.20	0.17	0.18	0.22
100-Recent History	0.15	0.16	0.13	0.13	0.07	0.22	0.14	0.10	0.17

Table C.15: Classifier Scores for Data-set: 90 days before admission with BOW representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.09	0.00	0.08	0.06	0.00	0.04	0.07	0.00	0.05
2500-Recent History	0.16	0.25	0.00	0.09	0.02	0.00	0.12	0.03	0.00
1500-Full History	0.15	0.00	0.24	0.04	0.00	0.11	0.06	0.00	0.15
1500-Recent History	0.11	0.00	0.04	0.13	0.00	0.02	0.12	0.00	0.03
1000-Full History	0.14	0.62	0.21	0.07	0.09	0.11	0.10	0.16	0.14
1000-Recent History	0.07	0.14	0.08	0.11	0.02	0.02	0.09	0.03	0.03
750-Full History	0.11	0.44	0.26	0.15	0.07	0.11	0.12	0.13	0.16
750-Recent History	0.08	0.20	0.08	0.11	0.02	0.11	0.09	0.03	0.10
500-Full History	0.12	0.40	0.07	0.06	0.15	0.02	0.07	**0.22**	0.03
500-Recent History	0.10	0.15	0.12	0.13	0.06	0.04	0.11	0.08	0.06
250-Full History	0.30	0.24	0.27	0.17	0.17	0.11	**0.21**	0.20	0.16
250-Recent History	0.17	0.25	0.25	0.19	0.15	0.11	**0.18**	**0.19**	0.15
100-Full History	0.28	0.23	0.31	0.15	0.22	0.15	0.19	0.22	0.20
100-Recent History	0.21	0.11	0.21	0.15	0.15	0.33	0.17	0.12	**0.26**

Table C.16: Classifier Scores for Data-set: 90 days before admission with Tf-idf representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.15	0.11	0.16	0.18	0.12	0.14	0.16	0.12	0.15
2500-Recent History	0.12	0.18	0.00	0.04	0.08	0.00	0.06	0.11	0.00
1500-Full History	0.13	0.12	0.16	0.16	0.12	0.22	0.14	0.12	0.19
1500-Recent History	0.11	0.15	0.38	0.08	0.08	0.06	0.09	0.10	0.10
1000-Full History	0.15	0.14	0.17	0.18	0.14	0.25	0.16	0.14	0.20
1000-Recent History	0.10	0.16	0.19	0.04	0.10	0.06	0.06	0.12	0.09
750-Full History	0.14	0.11	0.13	0.31	0.12	0.27	0.19	0.12	0.18
750-Recent History	0.16	0.17	0.15	0.10	0.12	0.08	0.12	0.14	0.10
500-Full History	0.11	0.15	0.14	0.27	0.14	0.33	0.16	0.14	0.20
500-Recent History	0.15	0.15	0.21	0.14	0.08	0.10	0.14	0.10	0.13
250-Full History	0.10	0.20	0.18	0.04	0.16	0.18	0.06	**0.18**	0.18
250-Recent History	0.16	0.15	0.14	0.06	0.08	0.10	0.09	0.10	0.12
100-Full History	0.15	0.22	0.21	0.22	0.08	0.20	0.18	0.12	**0.20**
100-Recent History	0.11	0.00	0.11	0.06	0.00	0.10	0.08	0.00	0.11

Table C.17: Classifier Scores for Data-set: 180 days before admission with BOW representation

Top-N words,	Precision			Recall			F1 score		
Full/Recent Data	LR	RF	SVM	LR	RF	SVM	LR	RF	SVM
2500-Full History	0.06	0.00	0.29	0.02	0.00	0.08	0.03	0.00	0.12
2500-Recent History	0.19	0.00	0.10	0.10	0.00	0.02	0.13	0.00	0.03
1500-Full History	0.22	0.00	0.24	0.08	0.00	0.08	0.12	0.00	0.12
1500-Recent History	0.15	0.50	0.00	0.10	0.02	0.00	0.12	0.04	0.00
1000-Full History	0.24	0.25	0.21	0.12	0.04	0.10	0.16	0.07	0.13
1000-Recent History	0.08	0.33	0.08	0.08	0.04	0.02	0.08	0.07	0.03
750-Full History	0.25	0.20	0.36	0.08	0.04	0.10	0.12	0.07	0.15
750-Recent History	0.10	0.00	0.07	0.10	0.00	0.04	0.10	0.00	0.05
500-Full History	0.21	0.27	0.25	0.14	0.08	0.14	0.16	0.12	0.18
500-Recent History	0.07	0.14	0.00	0.06	0.04	0.00	0.06	0.06	0.00
250-Full History	0.15	0.22	0.10	0.27	0.12	0.25	**0.19**	0.15	0.15
250-Recent History	0.11	0.16	0.12	0.14	0.08	0.16	0.12	0.11	0.14
100-Full History	0.12	0.09	0.12	0.20	0.10	0.16	0.15	0.10	0.13
100-Recent History	0.12	0.13	0.14	0.20	0.20	0.29	**0.15**	**0.16**	**0.19**

Table C.18: Classifier Scores for Data-set: 180 days before admission with Tf-idf representation

Appendix D

Parameters for Best Classifiers after Fine-tuning

Here, we provide the parameter settings for the best logistic regression, support vector machine and random forest classifiers when used on the embeddings. The best models using full and the past month's history for both types of embeddings follow below.

D.1 Classifiers after Regular Fine-tuning

Dataset (Days before admission)	LR	SVC	RF
0	C:1.06, class weights:balanced , penalty: l1	C: 0.56, class weights:(1:2)	estimators: 311, max depth: 29 , class weights: balanced
3	C: 1.15 , class weights: (1:10), penalty: l2,	C: 3.11 , class weights:balanced	estimators: 1155, max depth: 11, class weights: none
7	C: 1.68, class weights: (1:2), penalty: l1	C: 0.29 , class weights: balanced	estimators:311, max depth: 29, class weights: balanced
10	C: 0.0005, class weights: (1:5), penalty: l2	C: 0.37, class weights: (1:10)	estimators: 2000 , max depth: 8, class weights: balanced
14	C: 0.75, class weights: (1:20), penalty: l1,	C: 0.34 , class weights: (1:20)	estimators: 100 , max depth: 32, class weights: none
30	C: 0.59, class weights: none, penalty: l2	C: 0.11, class weights:(1:2)	estimators:1155, max depth:11 , class weights:none
60	C:0.75, class weights: (1:20), penalty: l1,	C: 0.41, class weights:balanced	estimators: 311, max depth:36 , class weights:none
90	C: 0.0005, class weights:(1:5) , penalty: l2	C: 0.37, class weights: (1:10)	estimators:311 , max depth:29 , class weights:balanced
180	C:0.0005 , class weights: (1:5), penalty: l2	C:0.41 , class weights:balanced	estimators:311, max depth:36 , class weights:none

Table D.1: Model Parameters: Best Classical Machine Learning Models on Embeddings after Regular Fine-tuning (Full History)

Dataset (Days before admission)	LR	SVC	RF
0	C: 0.16, class weights: (1:2), penalty: l2	C: 0.11 , class weights: (1:2),	estimators: 2000 , max depth:8 , class weights: balanced
3	C: 0.16 , class weights: (1:2), penalty: l1	C: 2.29, class weights: balanced	estimators:2000, max depth: 8, class weights: balanced,
7	C: 0.0005 , class weights: (1:5), penalty: l2,	C:0.29, class weights:balanced	estimators:311 , max depth:18, class weights: (1:20)
10	C:0.0005, class weights: (1:5), penalty: l2	C: 0.29, class weights: balanced	estimators: 311 , max depth: 18 , class weights: (1:20)
14	C: 0.0005, class weights: (1:5), penalty: l2	C:1.72, class weights: none	estimators: 100 , max depth: 32, class weights: none
30	C: 0.0005, class weights: (1:5), penalty: l2	C:0.52 , class weights:(1:20)	estimators:311 , max depth: 29, class weights:balanced,
60	C:0.0005 , class weights:(1:5) , penalty: l2	C:0.41 , class weights:balanced	estimators:1788 , max depth:25 , class weights:none
90	C:0.0005 , class weights:(1:5), penalty: l2	C:2.46, class weights:(1:10),	estimators:1155 , max depth: 11, class weights:none
180	C: 0.0005, class weights: (1:5), penalty: l2	C: 0.41, class weights:balanced	estimators: 100, max depth:32 , class weights:none

Table D.2: Model Parameters: Best Classical Machine Learning Models on Embeddings after Regular Fine-tuning (Past Month History)

D.2 Classifiers after Siamese Fine-tuning

Dataset (Days before admission)	LR	SVC	RF
0	C:0.0005 , class weights:(1:5) , penalty: l2,	C: 0.41, class weights:balanced	estimators: 1788, max depth: 25 , class weights: none,
3	C: 0.0005 , class weights: (1:5), penalty: l2,	C: 2.50 , class weights:balanced	estimators: 311, max depth: 29, class weights: balanced,
7	C: 1.06, class weights: balanced, penalty: l1,	C: 0.37 , class weights: (1:10)	estimators:944 , max depth: 36, class weights: none,
10	C: 1.06, class weights: balanced, penalty: l1,	C: 0.39 , class weights: (1:20)	estimators: 311 , max depth: 15, class weights: none,
14	C: 0.75, class weights: (1:20), penalty: l1,	C: 0.29 , class weights: balanced,	estimators: 311 , max depth: 29 , class weights: balanced
30	C: 1.16, class weights: (1:5), penalty: l1,	C: 1.55, class weights:(1:10)	estimators:1366, max depth:39 , class weights:(1:2)
60	C:0.75, class weights: (1:20), penalty: l1,	C: 0.29, class weights:balanced	estimators: 100, max depth:32 , class weights:none
90	C: 0.75, class weights:(1:20) , penalty: l1,	C: 2.90, class weights: (1:10)	estimators:311 , max depth:29 , class weights:balanced
180	C:1.06 , class weights: balanced, penalty: l1,	C:0.34 , class weights:(1:20)	estimators:311, max depth:29 , class weights:balanced

Table D.3: Model Parameters: Best Classical Machine Learning Models embeddings after Siamese Fine-tuning (Full History)

Dataset (Days before admission)	LR	SVC	RF
0	C: 1.16, class weights: (1:5), penalty: l1	C: 1.65 , class weights: (1:5),	estimators: 2000 , max depth:8 , class weights: balanced
3	C: 1.16 , class weights: (1:5), penalty: l1	C: 2.29, class weights: none,	estimators:311, max depth: 29, class weights: balanced,
7	C: 0.0005 , class weights: (1:5), penalty: l2,	C:1.16, class weights:(1:5)	estimators:311 , max depth:29, class weights: balanced,
10	C: 1.06, class weights: balanced, penalty: l1	C: 0.37, class weights: (1:10),	estimators: 2000 , max depth: 8 , class weights: balanced,
14	C: 1.06, class weights: balanced, penalty: l1,	C:0.29, class weights: balanced,	estimators: 2000 , max depth: 8, class weights: balanced,
30	C: 1.06, class weights: balanced, penalty: l1,	C:0.29 , class weights:balanced,	estimators:311 , max depth: 29, class weights:balanced,
60	C:3.83 , class weights:balanced , penalty: l1,	C:3.39 , class weights:(1:20),	estimators:100 , max depth:32 , class weights:none
90	C:0.75 , class weights:(1:20), penalty: l1	C:0.34 , class weights:(1:20),	estimators:311 , max depth: 29, class weights:balanced
180	C: 3.83, class weights: balanced, penalty: l1,	C: 0.41, class weights:balanced	estimators: 2000, max depth:8 , class weights:balanced

Table D.4: Model Parameters: Best Classical Machine Learning Models on embeddings after Siamese fine-tuning (Past Month History)