CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

# Predicting patient outcome from clinical journals and biomedical articles

## Using the MIMIC-IV database, multiple in-hospital mortality prediction models are created, to which improvements are attempted through the use of word embeddings trained on scientific biomedical literature

Master's thesis in Computer science and engineering

Fannie Henriksson
Philip Svensson

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

# Predicting patient outcome from clinical journals and biomedical articles

Using the MIMIC-IV database, multiple in-hospital mortality prediction models are created, to which improvements are attempted through the use of word embeddings trained on scientific biomedical literature

Fannie Henriksson, Philip Svensson

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

Predicting patient outcome from clinical journals and biomedical articles
Using the MIMIC-IV database, multiple in-hospital mortality prediction models are created, to which improvements are attempted through the use of word embeddings trained on scientific biomedical literature

Fannie Henriksson, Philip Svensson



Supervisor: Krasimir Angelov, Department of Computer Science and Engineering
Advisor: Magnus Kjellberg, Sahlgrenska University Hospital
Examiner: Nir Piterman, Department of Computer Science and Engineering

Master's Thesis 2022
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2022

Predicting patient outcome from clinical journals and biomedical articles
Using the MIMIC-IV database, multiple in-hospital mortality prediction models are created, to which improvements are attempted through the use of word embeddings trained on scientific biomedical literature

Fannie Henriksson, Philip Svensson
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

# Abstract

There exists an immense amount of data within the medical and healthcare field. Two examples of these are the patient's clinical records and biomedical research literature. The clinical records often contain numerical information suitable for developing AI-driven systems to support medical workers, and with some work, we believe that data from the biomedical literature can also be used to improve these systems.

In this work, clinical records from ICU patients in the MIMIC-IV database are used to create multiple prediction models to predict the risk of a subject passing away during their hospital stay (in-hospital mortality prediction). From the clinical records, abnormal observations are categorised into medical terms which are then used to find relevant biomedical research work from the PubMed database. This data is used to train word embeddings using Word2Vec. The word vectors for these categorised observations are then added as a concatenation to the clinical records data, in hopes of improving the predictions.

The results show very little difference with the additional information from the word embeddings. These are small improvements but unfortunately, we can not conclude that this can help improve such prediction models. We do however believe it suggest that there is valuable information that can be extracted from biomedical articles and that further work could show this.

Keywords: AI, ML, NLP, healthcare, data engineering, Word2Vec, MIMIC-IV

# Acknowledgements

We would like to thank our supervisor Krasimir Angelov and examiner Nir Piterman for making this thesis possible. We would also give a special thanks to Magnus Kjellberg, for your incredible support during the project.

Fannie Henriksson, Philip Svensson, Gothenburg, February 2022

# Contents

# List of Figures

# List of Tables

# 1
# Introduction

In recent years there have been great advancements within the healthcare sector, a large part of this is thanks to the modernisation and digitalisation of many parts of it. In recent years, the general use of AI and ML techniques has been getting traction and is a large contributing factor to these improvements in healthcare quality and efficiency. Some specific areas where this has been used with great success are within diagnostic disciplines such as automatic analysis of X-ray images [6]. It has also been used to predict diagnoses such as sepsis, which can have very large impact on mortality and readmission rates [7].

An extremely valuable source of information is the clinical records for each patient. These are documents that contain an immense amount of information about a patient and the interactions with them. This could include different types of measurements, diagnoses, interventions, etc. When extracted, this information can be used in creating prediction models for different patient outcomes. Such models can provide a basis for generating decision support systems that could aid medical workers in their clinical work.

Another valuable source of information is scientific biomedical literature. This is another way of studying how different diagnoses relate to patient outcomes which could also be used in creating prediction models. However, this often comes in the form of free text which might make the processing of the data much harder of a task, before it can be used to train a model. A great source for this information is PubMed, a database containing lots of biomedical studies and different types of health-related scientific studies.

## 1.1 Problem

In the clinical records of patients, many measurements and readings are well documented, for example, oxygen saturation, heart rate, and blood pressure. These measurements are called lab results, they come in many different forms but are often numerical. While models to predict patient outcomes can be created using numerical values as training data, they could perhaps be improved by gaining more contextual information. Perhaps this improvement could be made by combining data from clinical records with the PubMed database. If we can find certain medical conditions in the data from the clinical record they can perhaps be used to search for further information in the biomedical literature. We hope that this extra

contextual information can be used to improve the prediction models.

## 1.2 Goals and delimitations

The overall goal of the work is to find out if we can combine lab results from clinical records with data from scientific biomedical literature in order to improve prediction models for different patient outcomes. We hope that the added contextual value from the biomedical literature will provide the improvements we look for. This goal can be broken up into three separate challenges.

The first part is to study the clinical records of patients to create a prediction model which looks at numerical lab results to predict patient outcomes. This will be used as a baseline for comparisons with future models that make use of the data from the biomedical literature. We then want to give these numerical values some more context by categorising the measurements. For example, blood pressure could be categorised into *hypotension*, *normal* or *hypertension*. We will then use the categorisations to make prediction models which will also be used for comparisons. These medical terms for abnormal values are then going to be used to look for valuable information from the biomedical literature.

The second part is to train word vectors on scientific biomedical literature. We hope that this can provide a large amount of contextual information to be used despite not having much medical knowledge. To get the raw text data we hope to be able to use the previously generated medical terms as keywords while searching the PubMed database.

The last part is to combine the previous two to create new prediction models which make use of the information extracted with the word vectors. We hope that the added information from the scientific literature will lead to prediction improvements, compared to just using the clinical records. The goal is to see if different data will yield different prediction results. Thus an important delimitation is that we do not aim to create the best model with the best accuracy, since this would most likely not fit within the time limit of the project.

# 2
# Theory

In this chapter will theory and concept be presented. That includes which database we use, how previous work has implemented a benchmark, and how to evaluate binary classification algorithms. Models and techniques for evaluation and validation of machine learning models will also be presented. This chapter will end with theory about how word embeddings works. All the information in this chapter is good to know if you want to understand this thesis.

## 2.1 MIMIC-IV v1.0

This is the main data set used in this work. MIMIC-IV (Medical Information Mart for Intensive Care-IV) is a relational database containing data of real patients medical stays at a medical center in Boston, Massachusetts [8]. This database was published on March 16, 2021, and is a newer version of MIMIC-III. Compared to its previous version, MIMIC-IV contains data from 10 000 more patients, it has fewer typos, and the data structure is updated a bit. The data in the database comes from patients who were put in Intensive Care Units (ICU) between the years 2008 - 2019. The database is split into three major parts: core, hosp, and ICU.

### 2.1.1 Core

The core section includes three tables: admissions, patients, and transfers. In the admissions table is information of the admission times, where the admission took place and what the type of the admission is, for example, elective, emergency unit observation, or an urgent admission. There is also information of the discharge time and if the patient dies there is information about the death time. Other information is insurance, language, marital status, and ethnicity. The patients table has information of gender, anchor age, anchor year, anchor year group, and date of death. With this information can the year of birth and age be calculated. The transfers table includes intime and outtime and to which care unit the patient is moved. In all of these three tables subject id which is unique for each patient, and hadm id which is unique for each hospitial admission for each subject are included.

### 2.1.2 Hosp

The hospital electronic health records are contained in the hosp section. The data contains 17 different tables that comes from both the hospital visit and outside

the hospital, such as personal information records, although most measurements are done during the hospital stay. The tables in hosp have information about, for example, laboratory measurements, microbiology cultures and, medication administration, and prescription, these and some more tables are also used for hospital billing and insurances. There are also a few help-files, for example, diagnoses code and drug codes to link information between the tables.

### 2.1.3 Intensive Care Unit, ICU

The ICU table has information about when patients have been enrolled and discharged from the ICU and what kind of health tests that have been done, for example, intravenous and fluid inputs, procedures, and other charted information. It also contains help-files to link information between the tables. All tables where events occur also have a stay id which is unique for each patient.

### 2.1.4 Ethical aspects

All the data in MIMIC-IV is de-identified, which means that patient identifiers are removed according to the Health Insurance Portability and Accountability Act (HIPAA) Safe Harbor provision [8]. Some examples of this are replacing patient identifiers with random ciphers, shifting dates and times randomly into the future. Though all the data for a single patient is correctly relative since a single date shift is added per 'subject_id'. Meaning that if a patient had two measurements that were taken two hours apart they will also appear with that time difference in the MIMIC data set. To get access to the data you have to complete the course "Data or Specimens Only Research" which teaches you about how to process potentially sensitive data. The course can be navigated to from the MIMIC-IV page on PhysioNets website [8].

## 2.2 Previous work

Earlier work has highlighted the difficulties in measuring the advancements within the field of applying machine learning to clinical data [9]. A large part of this problem they assume to come from the absence of proper community benchmarks. Such benchmarks can accelerate the progress in machine learning research by allowing more people to take part in the ongoing discussions of how to advance within the field.

There are also a lot of papers about the MIMIC data set and about word embeddings, but this thesis takes inspiration from especially one paper [9]. That paper is a benchmark for MIMIC-III data set, which will be described in the following sections.

### 2.2.1 MIMIC-III benchmarks

The MIMIC-III Benchmark project was our first thought to be a good starting point for this paper, which made it important to understand the general layout of their

project. They propose a public benchmark suite for four different clinical prediction tasks, described in section 2.2.1.2 - 2.2.1.5, as well as formulating a multitask learning problem that involves jointly learning all four prediction tasks simultaneously. They present different models using logistic regression and various types of LSTM networks. One week after the beginning of the project, MIMIC released a newer version of the data set, MIMIC-IV. The newer data set has more patients, fewer typos in the journals and the structure of the files and tables are divided in another way. Unfortunately, the code from the benchmark project is not directly compatible with the newer MIMIC-IV data set and seems to have been abandoned a few years ago so some updates had to be done to the code in order for it to run. That was the main reason to redo the benchmark, to fit our project. The code was also a bit unclear and difficult to follow, for example, it was not clear at first how the split between training-, validation- and test data.

Some important terminology needs to be explained before presenting the benchmark work: In the MIMIC data set patients are referred to as subjects. Each ICU stay for a patient is referred to as an episode. A clinical event is an individual measurement, observation, or treatment. For the task-specific data sets, a sample is referred to as an individual record processed by a machine learning algorithm. For phenotyping a sample consist of an entire ICU stay, for tasks requiring hourly predictions (Length-of-stay) this includes all events up to a certain time.

#### 2.2.1.1 Prepocessing of the data set

In the benchmark project, during the initial processing of the data, some parts are excluded. Any hospital admissions with multiple ICU stays or transfers between different ICU units or wards are excluded. The authors claim that this is to reduce the ambiguity of outcomes associated with hospital admissions rather than ICU stays. They also exclude patients younger than 18 due to substantial differences between adult and pediatric physiology. Events that cannot reliably be matched to an ICU stay are then filtered out, this removes around 45 million events.

For each episode, a time series of events is compiled. This contains measurements from a set of 17 variables from a predefined list. These variables are not directly from the MIMIC data set, instead, a variable is compiled from multiple MIMIC variables. For example, there are eight MIMIC variables for weight, some in different units, these are all converted to the same unit and gathered as one variable. All the variables can be found in "resources/itemid_to_variable_map.csv". Lastly, the data is split with 15% as test data.

#### 2.2.1.2 In-hospital mortality prediction

A binary classification whether the patient will pass away during their hospital stay, based on the first 48 hours of their stay. Obvious cases are excluded when the data is prepared, such as stays ending before 48 hours or ones which have no observations in the first 48 hours. In the paper, they used AUR-ROC and AUC-PR for evaluation.

#### 2.2.1.3 Decompensation prediction

There are many ways to define decompensation, in the benchmark it is formulated as a binary classification problem, where the target label indicates whether the patient dies within the next 24 hours. This binary label is then assigned at every hour, starting at the fourth hour after admission. This is in order to have enough data to make an accurate prediction. The prediction stops when the patient either dies or is discharged. In the paper they used AUC-ROC and AUC-PR to measure performance.

#### 2.2.1.4 Length-of-stay prediction

Length-of-stay (LOS) is the length of a patient's hospital stay, often these only predict a total LOS at the beginning of an admission. The benchmark instead continually predict the remaining LOS at each hour, once again starting at hour 4 in order to have enough data, and ending once the patient dies or is discharged. For an evaluation metric, they use the Mean Absolute Difference (MAD) which is a standard regression metric.

#### 2.2.1.5 Phenotype classification

Phenotyping is classifying which acute care conditions a patient has. The benchmark classifies 25 conditions common in ICUs, including 12 critical ones, such as respiratory failure and sepsis, eight chronic conditions, and 5 mixed (meaning they are recurring or chronic with periodic acute episodes). These are identified using the single-level definitions from the Health Cost and Utilization (HCUP) Clinical Classification Software (CCS). The MIMIC database uses ICD-9 codes, which stands for International Classification of Diseases, to identify diseases and phenotypes labels, the CCS groups these together into mutually exclusive, largely homogenous categories. The classification here is retrospective in the way that a full ICU stay is observed before a prediction of a patients diseases is made. This is due to an important limitation in the MIMIC data set, the disease labels (ICD-9 codes) do not come with a timestamp to identify when a patient was diagnosed or became symptomatic. They also state that "because diseases can co-occur, we formulate phenotyping as a multi-label classification problem". As a metric of accuracy they chose to use macro- and micro-averaged AUC-ROC, macro-averaged being the main score.

## 2.3 Models

There are a lot of different models, but this project will use Logistic Regression, Random Forest, and Support Vector Machine, which are all described in more detail in Section 2.3.1 - 2.3.3. The reason for choosing these are that they are well known, easy to implement and get started with.

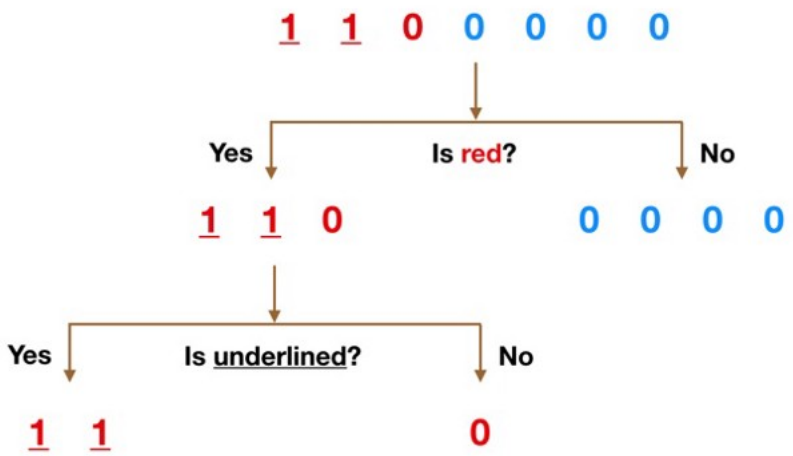


**Figure 2.1:** Example of a simple decision tree [1].

### 2.3.1 Logistic Regression

Logistic regression is a statistical model and has been commonly used in biological sciences [10]. The model tries to find a relation between a response variable and predictors, to calculate the probability that a certain event happens. For example, if the patient is to die in the ICU during the first 48 hours of their hospital stay. The response variable, $Y$, is binary and will only say yes or no. The predictors, independent variables, $X$, are for example the parameters, measurement values, from the patient journal.

### 2.3.2 Random Forest

Random forest is a commonly used classification algorithm. A random forest is built on uncorrelated decision trees. A decision tree consists of several data, first placed at the top, the root node. With a simple yes- or no question it will classify the data. For example, in Figure 2.1 you can see seven data points. First the tree asks if the data is red, since the four zeros are blue they will be classified as no, the rest as yes. Next question is, if they are underlined, which the ones are and thus classified as yes, and the zero is classified as no. The result is a prediction, one for yes and zero for no. The random forest consists of a lot of decision trees and together will the prediction be based on all the decision trees, a majority decision based on all decision trees. The algorithm is strong because the decision trees are uncorrelated to each other and protect each other from individual errors. To work, the random forest model needs to use the features better than random sampling, and the errors from each tree need to be uncorrelated or have low correlations with each other. To ensure the low correlation to each other, two methods are used, Bagging and Feature Randomness [1].

Decision trees are responsive to small changes in the training data, which can lead to the tree structures being different. The algorithm takes benefit of this, and bagging is when each tree randomly samples from the data set with replacement. Bagging

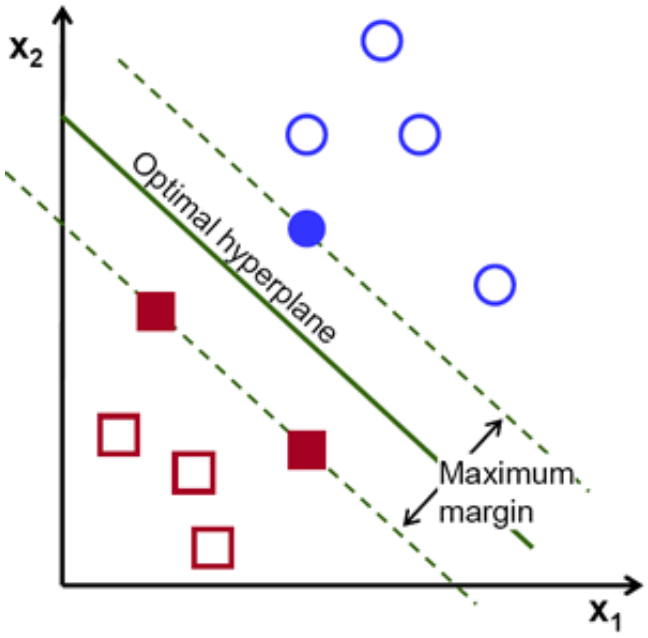


**Figure 2.2:** The optimal hyperplane of many hyperplanes [2].

improves the stability and accuracy of the algorithm and can also reduce variance and help to prevent overfitting [11].

When a decision tree splits a node, all the features are checked so that the algorithm can take the feature that makes the most separation between the data in the child nodes. When doing feature randomness, each tree can take from a subset of the features, and the subset is not the same for all trees.

### 2.3.3 Support Vector Machine, SVM

Support vector machine(SVM), is a supervised learning algorithm and can, for example, be used for classification, which is also known as Support Vector Classification(SVC). The algorithm tries to find the best hyperplane in an n-dimensional space, where n is the number of features. The best hyperplane is that plane that has maximum margin, the maximum distance between data points from each class, see Figure 2.2, which requires some reinforcement to classify incoming data points with better certainty. Hyperplanes are decision boundaries, and the output, which is a linear function, can be either 1 or -1, or two different classes [2].

## 2.4 Validation techniques for machine learning models with limited data

When fitting a machine learning model you want to ensure that it is stable. This means ensuring that patterns in the data set are recognised, and noise does not skew individual predictions. In other words, we want to maintain low bias and variance in order to create a model that is not underiftted or overiftted. This can be a

big issue if, for example, the data set used is too small for its intended purpose or contains too much of a class imbalance. There are various ways of identifying this and ensuring that the results you present are stable. This can be done by shuffling and re-sampling training and test data.

### 2.4.1 Cross-validation

Traditionally when training machine learning models one would split their data set into two or three parts. The most simple approach uses two splits, the first, called the training set, is often a large portion of the data set used to train the parameters of the model. The second set, which is often smaller, is then used to assess the performance of the model. This set is intentionally held out from the training of the model in order to ensure that the results are unbiased. Otherwise, the results would say very little about how the model performs on previously unseen data. Another approach is to use a third split which often is about the same size as the test data, this split is called the validation set. It is used to tune the hyperparameters of the model, these are the external parameters of the model which is different from the parameters estimated by the data set. Some examples of hyperparameters are: learning rate, the number of hidden layers in a neural network, or the number of estimators in the random forest algorithm.

Sometimes a single split could introduce a large variance in the results. This could happen if there is too little data in the test/validation set to contain all patterns present within the whole set. A way to solve this and check for variance within models trained on different data splits is to use cross-validation [12].

#### 2.4.1.1 K-fold cross-validation

For K-fold cross-validation the data set is divided into $K$ different subsets. These are then used to train the desired model multiple times. Each subset is used once as test/validation data, and the rest are used as the training data, resulting in the model being trained a total of $K$ times. Looking at different metrics one can identify variance between different model fits and get an average performance which might represent a more realistic result. Most commonly the value of $K$ is set to 10 [13].

#### 2.4.1.2 Stratified K-fold cross-validation

A problem that may arise during the development and training of machine learning models is that the dataset used contains class imbalances. For example, there might exist a lot of negative cases but very few positive ones in the data you are using. This could result in a variety of issues, for example, the ratio between classes in the training set might differ a lot from the ratio in the test set if the split it arbitrary. This may cause the evaluation on the test set to not be fair, in some sense, due to the test set having a much different distribution of classes, not representative of the original distribution [13]. To avoid this issue the data can be split in a stratified manner, which means that the original distribution is preserved for each subset. This is often used in cross-validation if some sort of class imbalance is known.

## 2.5 Evaluation of binary classification algorithms

There are many different ways to evaluate the performance of machine learning models and one has to be careful of how they interpret results when dealing with imbalanced data sets. For example, a model tested on a data set with 50% positive and 50% negative cases might achieve 90% accuracy, which could be a great result depending on the task. But if we instead look at a data set with only 10% positive cases and 90% negative cases, a bad model might not identify any patterns in the data and just predict according to the class distribution. Meaning that predictions are done at random with 90% being negative. This would result in an accuracy of 82%, which once again might sound great, but since the data set contains such imbalance it would mean that only 10% of positive cases would have been correctly identified. As one might see this presents some issues regarding how to interpret and present classification results in a way that is representative of how well the model actually performs. Therefore there exist many ways of evaluating the performance of machine learning models, such as binary classification models [14].

Performance metrics for evaluation of binary classifiers that are of special interest in this report are the confusion matrix, precision value, recall value, F-score, ROC(Receiver Operating Characteristic) curve and PRC(Precision-Recall Curve) [14].

### 2.5.1 Confusion matrix

This is one of the fundamentals of evaluation machine learning models. It contains all the predictions made and what category they fall into. For a binary classification problem the confusion matrix is 2x2 as can be seen in Table 2.1. The acronyms in the matrix are described as the following:
TN = True Negative, the number of negative values correctly classified as negative,
FN = False Negative, the number of positive values incorrectly classified as negative,
TP = True Positives, the number of positive values correctly classified as positive,
FP = False Positives, the number of negative values incorrectly classified as positive. The values in the confusion matrix are used as the basis for all other metrics presented here.

**Table 2.1:** Confusion matrix for a binary classifier

| | Predicted negative | Predicted positive |
|---|---|---|
| Actual negative | TN | FP |
| Actual positive | FN | TP |

### 2.5.2 Precision

Precision is used to measure what one could call correctness, i.e. it describes the certainty of a positive prediction being correct. It is often used when type 1 errors (false positives) have a high cost and want to be avoided. An example of this could

be an automated warning system where drastic measures might be taken once a warning is received. Here we obviously want a model that only warns when a threat actually is prevalent. Meaning that the precision of the model is high. As a numeric example, a precision of 0.95 would mean that every time the model identifies a threat, 95% of the time it is correct.

$$Precision = \frac{TP}{TP + FP} \tag{2.1}$$

### 2.5.3 Recall (or sensitivity)

Recall is used to measure the total amount of correctly identified positive cases. Since precision was used when type 1 errors had a high cost, recall is instead used when type 2 errors (false negatives) have a high cost and want to be avoided. An example of this could be a system where we want to classify whether or not a tumour is malignant or benign. Incorrectly labelling a tumour as benign when it is actually malignant could, in the worst case, lead to death. Obviously type 2 errors want to be avoided at all cost for such systems. As a numeric example, a tumour classifying model with a recall of 0.9 would mean that in 90% of all malignant tumours are correctly classified as such, and 10% are incorrectly classified as benign.

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

### 2.5.4 Traditional F-score and $\beta$ varied F-score

The traditional F-score, often referred to as the F1 score, F score, or F measure, is a harmonic mean of precision and recall. The F-score is proportional against the precision and recall and is used to get a more general idea of how a model performs on the positive classes since it combines both precision and recall.

$$F\text{-}score = \frac{2 * Recall * Precision}{Recall + Precision} \tag{2.3}$$

The traditional F-score is derived from a more general relationship between Recall and Precision, called the $\beta$ varied F-score. Here we can express which metric is more important. As a numerical example, to express that recall is more important than precision one can set $\beta = 2$ in Equation 2.4. If precision however weighs higher than recall, $\beta$ could instead be set to 0.5. This is useful for situations where misclassification among the minority class is expensive in general, and not just either among precision or recall. Looking back at the example of a model that identifies tumours as either malignant or benign, we might want a balance between the number of malignant tumours identified and the number of benign tumours misclassified as being malignant. But we much rather care about not missing any malignant ones compared to giving a false alarm on a tumour that is benign, this is then represented by a $\beta$ of less than 1.

$$F\text{-}score = \frac{(1 + \beta^2) * Recall * Precision}{\beta^2 Recall + Precision} \tag{2.4}$$

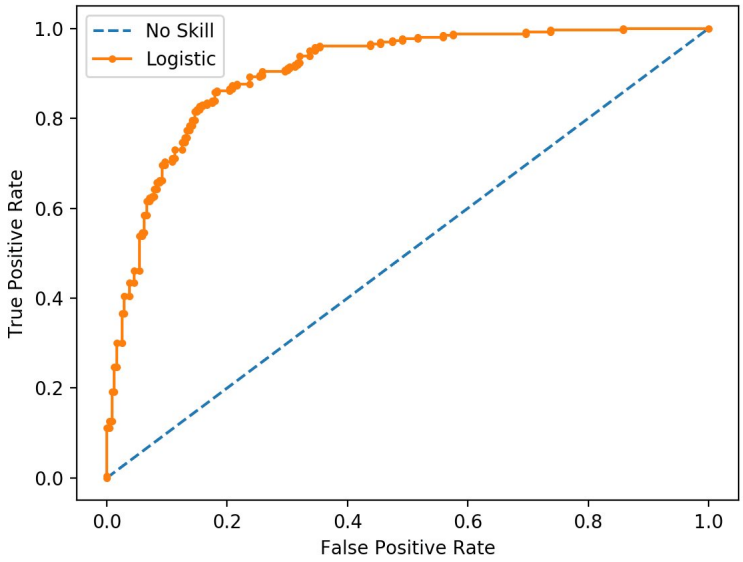


**Figure 2.3:** An example of a ROC curve [3].

### 2.5.5 ROC (Receiver Operating Characteristic)

The ROC curve is given by the true positive rate as a function of the false positive rate [15]. This is measured for a set of different probability thresholds. In other words, it gives an indication of how well the model can differentiate positive cases from negatives. This means that for a good performance model, it should be able to rank predictions in a way so that examples ranked higher are more likely to be positive. To evaluate this and plot the ROC curve for a prediction model we first begin by calculating the probabilities for all cases, this will be our ranking. How the probabilities are calculated are different depending on the algorithm chosen for the model. The list of probabilities is then sorted by their rankings, starting at the highest. To plot the curve we then go through the list of probabilities, if the case was positive we move one step in the y-direction and if it is negative we move one step in the x-directions, starting at (0,0). An example of a ROC curve can be seen in Figure 2.3. What this means is that a perfect classifier should be able to rank its probabilities so that all positive cases are ranked higher than any negative ones. Meaning the ROC curve will go straight up to (0,1) and then continue horizontally to (1,1). For a bad classifier the probabilities will be completely random and thus form a straight line from (0,0) to (1,1), often called the "no skill" line. This line is the baseline for ROC curves and the further away a model's curve is from it, the better it is at differentiating between positive and negative ones. An important thing to note here is that the baseline always is the same even if the ratio of positive to negative samples changes. Think of a data set with only 10% positive samples, and a random classifier with a threshold of 0.5. As this achieves a 50% accuracy it will also produce a 50% false positive rate.

To further explain the idea of the ROC curve an example test set of 20 data points

| Inst# | Class | Score | Inst# | Class | Score |
|---|---|---|---|---|---|
| 1 | p | .9 | 11 | p | .4 |
| 2 | p | .8 | 12 | n | .39 |
| 3 | n | .7 | 13 | p | .38 |
| 4 | p | .6 | 14 | n | .37 |
| 5 | p | .55 | 15 | n | .36 |
| 6 | p | .54 | 16 | n | .35 |
| 7 | n | .53 | 17 | p | .34 |
| 8 | n | .52 | 18 | n | .33 |
| 9 | p | .51 | 19 | p | .30 |
| 10 | n | .505 | 20 | n | .1 |

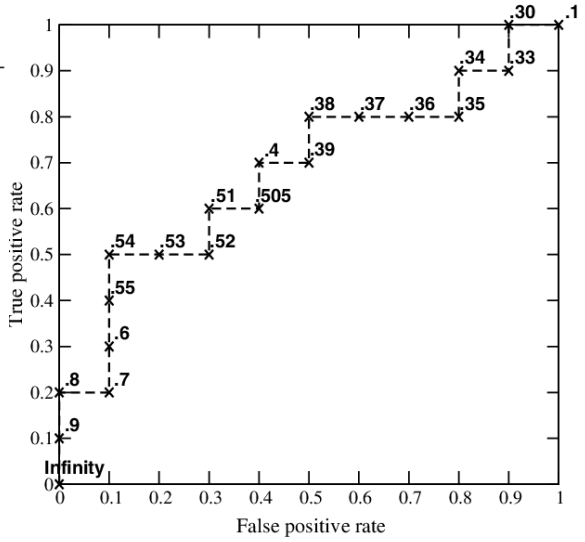


**Figure 2.4:** An example of a ROC curve created using threshold from a test set. The table shows the 20 data points, their true class label and their prediction score. The graph then shows the ROC curve plotted by these 20 points, each labelled by the threshold it produces.

and its corresponding ROC curve is shown in Figure 2.4 and further explained here. For this example we have 20 instances of data that will be classified. The ground truth variable for each instance is denoted by either p (positive) or n (negative). The classification here is binary but the prediction itself is a numeric probability score between zero and one. What then decides whether a prediction is positive or negative is the chosen probability threshold, this is a numerical threshold for when a prediction should be classified as positive or negative. E.g. a threshold of 0.5 would mean that all predictions with a probability score greater than, or equal to, 0.5 should be classified as positive, otherwise it is classified as negative. These probability scores can be calculated from many types of different algorithms, for example logistic regression and random forest, but for this example they are just arbitrary numbers, for the sake of an example. With each instance given a probability score, they are then sorted from highest score to the lowest, this means that the instances deemed to most likely be positive will be first in the list, and the ones most likely to be negative will be at the bottom. To calculate the ROC curve we then proceed according to the instruction from the previous paragraph. We begin at the highest-ranked instance, meaning the one given the highest probability score and thus most likely to be positive. Since this instance indeed was positive we take one step in the y-axis on the plot, starting at (0,0), as can be seen in Figure 2.4. The second instance is also positive so we take another step in the y-direction. Looking at the third step however we can see that it is negative, this means that we instead take a step along the x-axis. As we can see here this means that if all positive cases have higher scores than any negative ones the ROC curve would be represented by a straight line from (0,0), to (0,1) and then to (1,1), as previously described. As we can see here, generally the closer the curve is to the (1,1) corner, the better it is. In the previous paragraph we describe the baseline, or the "no skill" line, as the

line that goes straight from (0,0) to (1,1). This is indicative of a random classifier which does not know anything of the class an instance belong to. Take the example in Figure 2.4, if the classifier would be completely random the probability score would also be random, resulting in the list of instances being shuffled with positive and negative instances almost perfectly alternated. Looking at the previous example we can see that this would result in a line from (0,0) to (1,1), the baseline that is.

To measure the difference from the baseline we can calculate the AUC. This gives a quantitative way of telling how far from the "no skill" line a certain model is, it also allows us to compare the performance of different models to each other. Another use of the ROC curve is to find the optimal prediction threshold. That is, the probability for which all values above it should be predicted as "True" and all values below it as "False". For example, a threshold of 0.55 would mean that if a prediction is given a probability above 0.55 it is classified as positive, otherwise it is classified as negative. The optimal threshold would be represented by the point on the curve furthest away from the "no skill" line. Optimal in the sense here means the threshold that gives the best trade-off between true positive and false positive rate.

### 2.5.6 PRC (Precision Recall Curve)

The PRC is much like a ROC curve but instead of the positive rate as a function of the negative rate, it shows the trade-off between precision and recall for different thresholds. An example of a PRC can be seen in Figure 2.5. This is often used instead of the ROC curve if the data set is heavily imbalanced, meaning that negative cases are dominating [16]. Instead of starting at (0,0), a PRC starts at (0,1). A perfect model would be indicated by a curve going from (0,1) to (1,1) and then straight down to the baseline, indicating both perfect precision and recall. An important thing to note here is that the baseline curve no longer is the same for all ratios of positive and negative samples. For a PRC the baseline is calculated from the ratio of positive to negative samples, $y = P/(P + N)$. For a data set with only 10% positive samples, this would correspond to a baseline at $y = 0.1$. This makes sense since a random classifier that has a threshold of 0.5 would end up with a recall of 0.5 and a precision of 0.1, which corresponds to a point at the line $y = 0.1$. Note that this also means the AUC has to be interpreted with the baseline in mind, as a higher baseline naturally would give a higher AUC.

### 2.5.7 How these methods will be used for this project

The evaluation methods described in this section will all be used during the evaluation of the machine learning models created in this project. All of them except two will be used directly, the others are used indirectly as described throughout Section 2.5. The confusion matrix is used indirectly as it is used to understand and calculate some of the other measurements. The $\beta$ varied F-score is also not directly used, it is only used to help understand the traditional F-score, which is derived from the more general version. The evaluation methods will be used for all algorithms described in Section 2.3. Results of these can be seen in Section 4.1.1, 4.1.2 and 4.1.3.

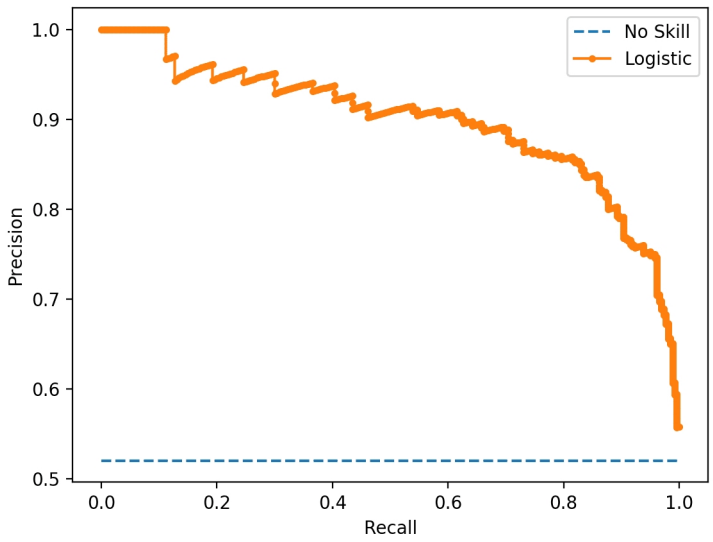


**Figure 2.5:** An example of a PRC [3].

## 2.6 Interquartile Range Method

When working with large databases there are bound to be some errors, these can come from human factors or something like a broken sensor recording inaccurate measurements. These are known as outliers and should preferably be removed in order to not distort the data. Interquartile range (IQR) is a measure of statistical dispersion, and the interquartile range method is one way of identifying these outliers. This method is good when the data is not good enough to be treated as a Gaussian distribution. To calculate the IQR, take the difference between the 75th and 25th percentile, see Equation 2.5 [17].

Percentile is a value that divides a data set into a certain percent. For example, the 50th percentile is the same value as the median. Below the value is 50% of the values in the data set, and that means also the other 50 % of values in the data set is over the percentile value. When the data is divided into 25th, 50th, and 75th percentile, the percentile often refers to quartiles, where the 25th percentile is the same as Q1, and the 75th percentile is the same as Q3 [18].

To do the interquartile range method, IQR is multiplied with a factor K, called cut-off value, Equation 2.6. Then the limits for what is an outlier or not is the cut-off value below Q1 and above Q3, see equation 2.7. The value of K is commonly set to 1.5. A boxplot use the interquartile range method, shown in figure 2.6.

$$\text{IQR} = 75\text{th percentile} - 25\text{th percentile} \tag{2.5}$$

$$\text{cut off} = \text{IQR} * K \tag{2.6}$$

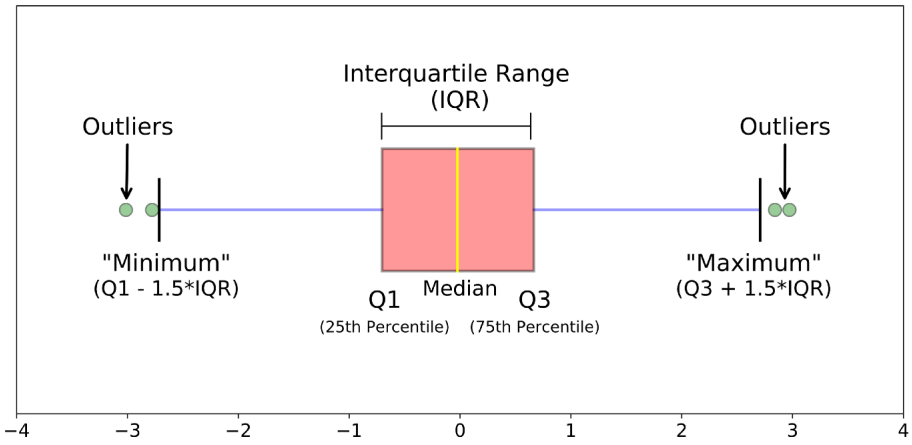


**Figure 2.6:** An illustration of how IQR is used in a boxplot [4].

$$\begin{aligned} &\text{outliers} < 25\text{th percentile} - \text{cut off} \\ &\text{outliers} > 75\text{th percentile} + \text{cut off} \end{aligned} \tag{2.7}$$

## 2.7 Natural language processing and word embeddings

Natural Language Processing (NLP) is a subfield within computer science regarding linguistics and artificial intelligence. It revolves around how computers process and interpret natural language data of any size, all the way from large corpora to individual words. Computers are well known to be good at handling numerical data, but when it comes to processing natural language data it might not be as obvious how to tackle the problem. The medical area is an area where NLP could have a large impact, lots of scientific biomedical data from reports and research papers exists, on sites such as PubMed, which could be analysed using NLP [19]. Often such texts require expertise within the field they are written in, but enabling computers to do the processing of the texts makes it easier for people with less experience within the field to work with them. Though it can still be of great value to those who are experts on the topic.

### 2.7.1 Word2Vec

A common way of analysing natural language is through the use of word embeddings, which is a way of representing words as numerical vectors. This makes for a natural way of comparing words since basic arithmetic operations, such as subtraction and addition, now can be applied to them. For example, words could be combined or subtracted to generated a new word, a common example to illustrate this is the following: *king* − *man* + *woman* = *queen*. What this means is that we should be able to add the vectors associated with *king* and *woman*, subtract the vector for *man*

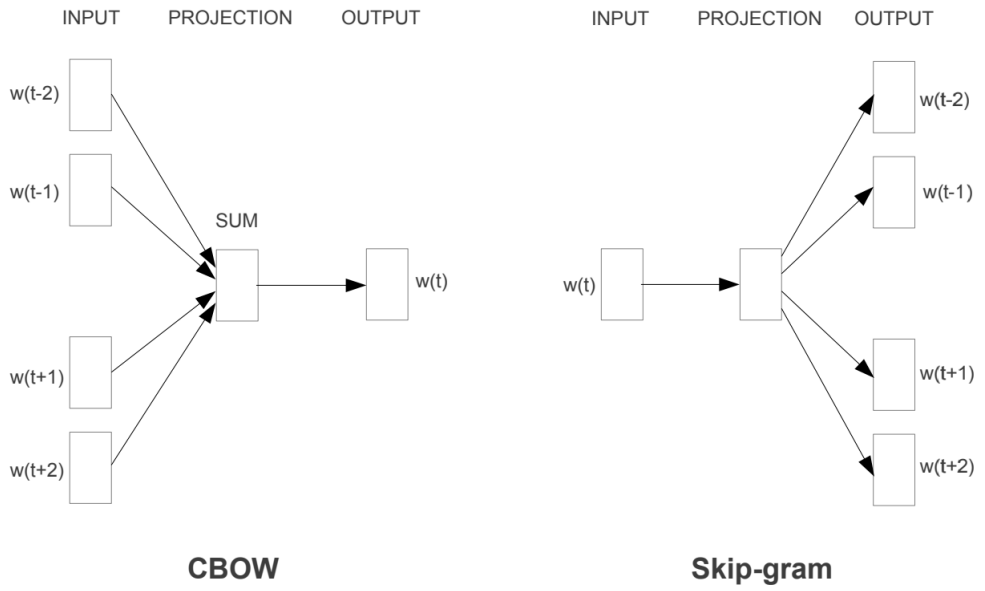


**Figure 2.7:** An illustration of how CBOW and Skip-gram works [5].

to get a resulting vector representation of the word *queen*. A technique that makes use of word embeddings is Word2Vec, which was created, patented and published in 2013 [20] [5].

Word2Vec can be used to train vector spaces from hundreds of millions of words, which it can do using two different techniques called Continous Bag Of Words(CBOW) and Skip-gram. These both consist of a neural network with a single hidden layer, after the training is done, the weights of the hidden layer are then used to represent a words embedding. How the training is done differs between the techniques however. CBOW predicts the target word from a window of surrounding context word, meaning that the input to the network consist of multiple words and the target is a single word. Skip-gram does the opposite, it predicts a window of context words given a single word, thus the input is a single word and the target consists of multiple words. An illustration of this can be seen in Figure 2.7. Inputs and targets are encoded as one-hot vectors, which is a vector where only one element is 1, the rest are 0. The input and target vectors are of size 1 x $n$, where $n$ is the number of unique words in the corpus used.

# 3
# Methods

This project creates various machine learning models to measure the improvements that word embeddings from biomedical literature can add to the task of predicting outcomes of patients in ICU. To achieve this, firstly a pipeline similar to the one described in Section 2.2.1 is created, which turns the data from the MIMIC-IV database into a format usable for machine learning and creates prediction models based on it. Compared to the previous work this project is built upon the newer MIMIC-IV database, which was released right at the beginning of the project, thus there were little to no other projects on this specific data set at the time. The second part of the project has nothing to do with the benchmark paper. It revolves around gathering abstracts from the PubMed database and using this to train word embeddings which can then be incorporated into the prediction models created from the previous step. All code in this project can be find at a GitHub repository (https://github.com/Phaxboi/predicting-patient-outcome).

## 3.1 Preparing the MIMIC data

Since the MIMIC-IV data set is a relational database with many different files the data does not come in a format that immediately is usable for the purposes of this project. Therefore some pre-processing of the data has to be done in order for it to be used in our models. The pre-processing is split into different steps that are all run separately, though they need to be run in sequential order to function properly. This approach has many benefits, but mainly two that made working with this data set easier. It makes debugging significantly easier, files can be compared before and after in order to minimise the risk of information being dropped or modified in the wrong way. Having the project being run in multiple separate modules also makes implementing changes less of an inconvenience, since only modules after the one being changed have to be re-run. Below are descriptions of the core parts of the data extraction.

Since what we want to achieve with the MIMIC-IV data is to format it as time series, we use the format of those from the benchmark paper and MIMIC-III as a guideline for how the result should look like. However due to the structural changes between MIMIC-II and MIMIC-IV the code itself from the benchmark can not be directly reused, instead new code which works with the newer database has to be created. Some general ideas could still be used however, the benchmark paper for example removes some stays on some criteria, which we also made use of, this is

described in the section below.

### 3.1.1 Extracting core patient information

The first step is to extract what we define as the core patient information, this is mostly gathered from 'core/patients' and 'core/admissions', but also from 'icu/icustays'. For each patient, a folder is created and named according to their subject id. Inside each folder, a file named 'patient_info_summary.csv' is created. This file contains a summary of information regarding each unique stay per patient, the information extracted is presented in Table 3.1. A summary of all patient's stays is also created as a single file called 'stays.csv', where each row represents a unique stay.

The MIMIC-IV database contains information about many patients and their stays but not all of them are of interest in this project, therefore a lot of stays and subjects have to be removed. Below are listed the criteria we filter along with the reasoning behind excluding them.

- **Age**: We remove patients younger than 18 due to differences in adult and pediatric physiology. The age at which patients should seek pediatric care varies a lot for different countries but 18 is commonly used. This also happens to be the same way it is done in the MIMIC-III benchmark paper [9].
- **Stays which start and end in different care units**: Since a patient might have been transferred to a non intensive care unit during their stay there might be a gap in their timeline, only data from ICU exists, thus a complete time series can not be recreated. Due to the uncertainty of what happens between the transfers, these are removed.
- **Stays that ends before 48 hours**: In order to have a substantial amount of data per stay we remove those ending before 48 hours, for further research this threshold could be moved.

After removing the stays we are left with 32 393 valid ones from 25 720 unique patients for us to look at. Out of these a total of 4487 have passed away at the hospital, 4480 of these have an in-hospital death time. In the case of a missing time, it can be obtained by either looking at the discharge time or calculated from the length of stay and the time of being transferred to ICU. Ideally the death time and discharge time should be the same, but sometimes they are not. Therefore missing death time values are replaced with the time at which a patient was transferred into the care unit plus the length of stay field. This seems to always add up to the death time value in the cases where they all are present.

Another important note is that we do not look at any data before the patient was admitted to an ICU. Information from previous ward stays can be obtained by looking at the 'core/transfers' table and matching upon *subject_id* and *hadm_id*. The transfers table then contains a field called *transfer_id* from which *stay_id* can be derived.

**Table 3.1:** The variables extracted for each unique stay.

| MIMIC table column name | Description | How it is obtained |
|---|---|---|
| subject_id | Subject identifier per patient | core/patients |
| gender | The gender of a patient | core/patients |
| age | The actual age of a patient | core/admissions/admittime - birth_year |
| birth_year | Anchor birth year | core/patients/anchor_year - patients/anchor_age |
| dod | Date of death | core/patients |
| hadm_id | Unique identifier per hospital admission | core/admissions |
| admittime | Time of admission to the hospital | core/admissions |
| dischtime | Time of discharge from the hospital | core/admissions |
| deathtime | In-hospital death time of patient | core/admissions |
| stay_id | Unique identifier per ward stay | icu/icustays |
| first_careunit | First ICU the patient was admitted to | icu/icustays |
| last_careunit | Last ICU the patient was admitted to | icu/icustays |
| intime | The time at which a patient was transferd into ICU | icu/icustays |
| outtime | The time at which a patient was transferd out of ICU | icu/icustays |
| los | Length of stay for the given ICU stay | icu/icustays |
| hospital_expire_flag | Indicates whether the patient died in-hospital | core/admissions |

### 3.1.2 Extracting events

The next step in the process is to extract various lab results and measurements, referred to as events, for each stay and put them together to form time series. For each patient, an 'events.csv' file is generated where all events for a patient are written. The fields extracted per event are listed in Table 3.2. Each event contains the three identifiers *subject_id*, *hadm_id* and *stay_id*, these identifies the unique subject, unique admission and unique stay. As well as the time when the measurement was made is recorded, along with the time of transfer to the ICU. Each event also has a unique MIMIC-IV specific variable used as an identifier for what was measured (eg. 212 corresponds to heart rate), along with the recorded value.

#### 3.1.2.1 Clinical variables/laboratory values to track

Before any of the data from the 'events.csv' file can be used there is some work that has to be done. As mentioned earlier each event is identified by an 'ITEMID' which is a MIMIC-specific variable that links to an actual description of the value eg. heartbeat, blood pressure, etc. However a single concept can have multiple MIMIC variables that correspond to it. For example, there exist two MIMIC variables for body temperature, one in Fahrenheit and one in Celsius each has a unique 'ITEMID'. Another example is capillary refill rate, which has two related variables 'Capillary refill L' and 'Capillary refill R'. This meant that in order to make this information useful new variables had to be created. The new variables map one or more MIMIC-specific variables to a single one. For exact details on the new variables and how they map to the MIMIC-specific variables see Table 3.3.

Here a problem arises where we do not have any medical expertise or knowledge in order to make qualified decisions to decide what variables are the best to track. Therefore, we consulted our supervisor who has some expertise within the area. It was then suggested that we went with the variables from the MIMIC-III benchmark paper [9]. 16 out of 17 variables from the paper were chosen as can be seen in the first column in Table 3.3. The variable left out is 'Glasgow coma scale total' because it only exists in MIMIC-III and not in MIMIC-IV. It is important to note that the only part that can be used from the benchmarks variables are their names. MIMIC-

**Table 3.2:** The information per event in the 'events.csv' file.

| MIMIC table column name | Description |
|---|---|
| subject_id | Subject identifier per patient |
| hadm_id | Unique identifier per hospital admission |
| stay_id | Unique identifier per ward stay |
| itemid | MIMIC specific variable used for identification of a measurement (eg 212 = heart rate) |
| intime | The time at which a patient was transferd into ICU |
| charttime | Records the time a measurement was made |
| value | Value of the measurement, can be numerical or a string, does not contain any unit of measurement |

III and MIMIC-IV do not have the same MIMIC specific variables and ITEMIDs. Though many of the MIMIC-III and MIMIC-IV variables are similar, the biggest difference is that multiple MIMIC-III variables are clumped into a single MIMIC-IV variable.

Some extra pre-processing of this data is also needed due to different reasons. As mentioned earlier some variables come in different units of measurement, weight comes in both kg and lbs, for all such variables we convert them to the same unit. Outliers also have to be removed, since in some rare cases there exist measurements that do not make any sense, either by being too high, too low, or perhaps negative in the case of a variable that should be positive. Since we do not have any medical expertise to determine what the exact cutoff points for outliers should be the Interquartile Range Method is used, explained in Section 2.6. A factor of 10 is used for this since it removes the worst outliers, while not removing values that are too close to being humanly possible for us to determine.

### 3.1.3 Creating time series

The data from the events are simply observations or measurements with an attached timestamp. A common way to present this data would be as a time series. For each episode, a time series is then created from all its corresponding events gathered. As mentioned earlier a unique stay is often referred to as an episode, a single patient can have multiple episodes if they were admitted to the hospital times. A time series is a table where each row is indexed by a time stamp. Each column then contains the measured values for a specific variable at that time, these variables are the new ones created shown in Table 3.3. If no value is present for a given timestamp that value is left empty, note that it is not replaced with a zero.

The time series previously mentioned consists of all events during a patient's stay. 48h time series are also created to define a time interval to be studied for the specific prediction task. This however poses an issue where some time series can have values measured with a negative timestamp. These are often things measured immediately when a patient arrives at the hospital, such as weight and length. Later during the stay these values are rarely measured, thus leaving many timestamps empty. Therefore when creating 48h time series they begin at the earliest time stamp up to 30 minutes before the admission time. This is in order to ensure that the beginning of a time series is not too sparse, but still keeps some of the early measurements.

**Table 3.3:** Our 16 variables and what MIMIC-IV variables they are mapped to.

| Variable | MIMIC-IV variable | Item ID | Unit of measurements | MIMIC-IV table |
|---|---|---|---|---|
| Capillary refill rate | Capillary Refill L | 224308 | | chartevents |
| Capillary refill rate | Capillary Refill R | 223951 | | chartevents |
| Diastolic blood pressure | Arterial Blood Pressure diastolic | 220051 | mmHg | chartevents |
| Diastolic blood pressure | Non Invasive Blood Pressure diastolic | 220180 | mmHg | chartevents |
| Diastolic blood pressure | Manual Blood Pressure Diastolic Left | 224643 | mmHg | chartevents |
| Diastolic blood pressure | Manual Blood Pressure Diastolic Right | 227242 | mmHg | chartevents |
| Diastolic blood pressure | ART BP Diastolic | 225310 | mmHg | chartevents |
| Fraction inspired oxygen | Inspired O2 Fraction | 223835 | None | chartevents |
| Glasgow coma scale eye opening | GCS - Eye Opening | 220739 | | chartevents |
| Glasgow coma scale motor response | GCS - Motor Response | 223901 | | chartevents |
| Glascow coma scale verbal response | GCS - Verbal Response | 223900 | | chartevents |
| Glucose | Glucose (serum) | 220621 | None | chartevents |
| Glucose | Glucose finger stick (range 70-100) | 225664 | None | chartevents |
| Glucose | Glucose (whole blood) | 226537 | None | chartevents |
| Heart rate | Heart Rate | 220045 | bpm | chartevents |
| Height | Height | 226707 | Inch | chartevents |
| Height | Height (cm) | 226730 | cm | chartevents |
| Mean blood pressure | Arterial Blood Pressure mean | 220052 | mmHg | chartevents |
| Mean blood pressure | IABP Mean | 224322 | mmHg | chartevents |
| Mean blood pressure | ART BP Mean | 225312 | mmHg | chartevents |
| Mean blood pressure | Non Invasive Blood Pressure mean | 220181 | mmHg | chartevents |
| Oxygen saturation | Arterial O2 Saturation | 220227 | % | chartevents |
| Respiratory Rate | Respiratory Rate | 220210 | insp/min | chartevents |
| Respiratory Rate | Respiratory Rate (spontaneous) | 224689 | insp/min | chartevents |
| Respiratory Rate | Respiratory Rate (Total) | 224690 | insp/min | chartevents |
| Respiratory Rate | Spont RR | 224422 | bpm | chartevents |
| Systolic blood pressure | Arterial Blood Pressure systolic | 220050 | mmHg | chartevents |
| Systolic blood pressure | ART BP Systolic | 225309 | mmHg | chartevents |
| Systolic blood pressure | Non Invasive Blood Pressure systolic | 220179 | mmHg | chartevents |
| Systolic blood pressure | Manual Blood Pressure Systolic Left | 224167 | mmHg | chartevents |
| Systolic blood pressure | Manual Blood Pressure Systolic Right | 227243 | mmHg | chartevents |
| Temperature | Temperature Fahrenheit | 223761 | °F | chartevents |
| Temperature | Temperature Celsius | 223762 | °C | chartevents |
| Weight | Daily Weight | 224639 | kg | chartevents |
| Weight | Admission Weight (Kg) | 226512 | kg | chartevents |
| Weight | Admission Weight (lbs.) | 226531 | lbs | chartevents |
| pH | PH (Arterial) | 223830 | None | chartevents |
| pH | PH (Venous) | 220274 | None | chartevents |

### 3.1.4 Categorising measured values

Before any meaningful data can be extracted from our word embedding model we need to know what words we want to look at. This poses a challenge for us to turn the numerical data into some words which can be used to extract further information about a patient's hospital stay and their health status. The first step in doing this is to categorise all the numerical values from the time series into medical terms that give a more contextual description of the medical status of a patient. For example blood pressure will be categorised into the three categories *hypotension*, *normal* or *hypertension*. Each category is also encoded by a numerical value, this can be seen as a discretisation of the original numerical measurements, and will be used to generate categorical features, described in Section 3.3.2. Each variable is categorised based on medical reference values regarding what is normal and abnormal. The categorisations and encodings are summarised in Tables 3.4 and 3.5 and further details regarding how they are obtained can be viewed in Section 3.1.4.1.

#### 3.1.4.1 Reference values used for categorisation

The reference values for height are based on the Anthropometric Reference Data for Children and Adults in the United States between 2003 and 2006 [21]. We look at the data of males and females age 20 and up. Short and tall stature are defined as two or more standard deviations below or above a populations mean, which means that around 5-6% of a population suffers from it [22] [23]. We do not differentiate between male and female patients when categorising measurements since we did not have time to rewrite the code to support it. However the mean height of males and females are quite different, so as a compromise we classify short stature as any patient having a lower height than what would classify as short stature among women, since they are in general shorter than men. Then we define tall stature as any patient being taller than what is classified as tall stature amongst men. Thus anyone who is classified as having tall- or short stature would have been classified as that regardless of their gender. Meaning a few will be missed, but we felt that this was the best compromise.

The reference values for mean blood pressure are derived from the reference values of diastolic blood pressure and systolic blood pressure [24] [25]. The lower bound for mean blood pressure, which signifies hypotension, is derived from the lower bound of diastolic blood pressure. The upper bound, which signifies hypertension, is then derived from the reference values of systolic blood pressure.

The Glasgow coma scale is used to check how awake a patient is [26]. The scale has three parameters: eye-opening, verbal response, and motor response. The Glasgow coma scale already uses a scale from one and upwards, so that can be used directly, see Table 3.5. If the reading is labelled as "not testable", we set the category to zero.

The reference values for the rest of the variables come from different health care institutes and researches. Capillary refill rate takes its values from Nursing Times

[27], fraction-inspired oxygen and pH are research articles [28] [29], and respiratory rate from Johns Hopkins Medicine [30]. Temperature and low blood pressure are taken from 1177 Vårdguiden [31] [32] [25]. Glucose is taken from diabetes.co.uk [33]. Heart rate, and oxygen saturation from Vårdhandboken [34] [35], and the reference values for high blood pressure are taken from Hjärt-Lungfonden [24]. Height values are taken from the CDCs National Health Statistics Reports [21]. All reference values, except for the Glasgow coma scale, are summarised in Table 3.4.

**Table 3.4:** Reference values and their associated medical terms.

| Categories | Reference values | Unit | Numerical values for the categories |
|---|---|---|---|
| Capillary refill rate | **Normal:** <= 2<br>**Abnormal:** > 2 | seconds | **Normal:** = 0<br>**Abnormal:** = 1 |
| Diastolic blood pressure | **Hypotension:** < 60<br>**Normal:** 60 - 90<br>**Hypertension:** > 90 | mmHg | **Hypotension:** = 0<br>**Normal:** = 1<br>**Hypertension:** = 2 |
| Fraction inspired oxygen | **Low:** < 21<br>**Normal:** = 21<br>**High:** > 21 | % | **Low:** = 0<br>**Normal:** = 1<br>**High:** = 2 |
| Glucose | **Hypoglycemia:** < 72<br>**Normal:** 72 - 200<br>**Hyperglycemia:** > 200 | mg/dL | **Hypoglycemia:** = 0<br>**Normal:** = 1<br>**Hyperglycemia:** = 2 |
| Heart rate | **Bradycardia:** < 50<br>**Normal:** = 50 - 100<br>**Tachycardia:** > 100 | bpm | **Bradycardia:** = 0<br>**Normal:** = 1<br>**Tachycardia:** = 2 |
| Height | **Short stature:** < 139.9<br>**Normal:** 139.9 - 199.3<br>**Tall stature:** > 199.3 | cm | **Short stature:** = 0<br>**Normal:** = 1<br>**Tall stature:** = 2 |
| Mean blood pressure | **Hypotension:** < 60<br>**Normal:** 60 - 140<br>**Hypertension:** > 140 | mmHg | **Hypotension:** = 0<br>**Normal:** = 1<br>**Hypertension:** = 2 |
| Oxygen saturation | **Hypoxemia:** < 90<br>**Normal:** 90<br>**Hyperoxia:** > 90 | % | **Hypoxemia:** = 0<br>**Normal:** = 1<br>**Hyperoxia:** = 2 |
| Respiratory Rate | **Bradypnea** < 12<br>**Normal:** 12 - 16<br>**Tachypnea:** > 16 | insp/min | **Bradypnea:** = 0<br>**Normal:** = 1<br>**Tachypnea:** = 2 |
| Systolic blood pressure | **Hypotension:** < 90<br>**Normal:** 90 - 140<br>**Hypertension:** > 140 | mmHg | **Hypotension:** = 0<br>**Normal:** = 1<br>**Hypertension:** = 2 |
| Temperature | **Hypothermia:** < 35<br>**normal:** 35 - 37.8<br>**Hyperthermia:** 37.8 - 40<br>**Hyperpyrexia:** > 40 | °C | **Hypothermia:** = 0<br>**Normal:** = 1<br>**Hyperthermia:** = 2<br>**Hyperpyrexia:** = 3 |
| Body Mass Index<br>BMI | **Underweight:** < 18.5<br>**Normal:** 18.5 - 25<br>**Overweight:** 25 - 30<br>**Obesity:** > 30 | | **Underweight:** = 0<br>**Normal:** = 1<br>**Overweight:** = 2<br>**Obesity:** = 3 |
| pH | **Acidemia:** < 7.35<br>**Normal:** 7.35 - 7.45<br>**Acidosis:** > 7.45 | | **Acidemia:** = 0<br>**Normal:** = 1<br>**Acidosis:** = 2 |

## 3.2 Word embeddings

The progress of creating word embeddings consists of two major parts. The first involves gathering large amounts of text data, in this case the data will come from

**Table 3.5:** Glasgow coma scale

| Categories | Numerical values for the categories |
|---|---|
| Glasgow coma scale eye opening | **Not Testable** = 0<br>**Does not open eyes** = 1<br>**Opens eyes in response to pain** = 2<br>**Opens eyes in response to voice** = 3<br>**Opens eyes spontaneously** = 4 |
| Glasgow coma scale motor response | **Not Testable** = 0<br>**Makes no movements** = 1<br>**Extension to painful stimuli** = 2<br>**Abnormal flexion to painful stimuli** = 3<br>**Flexion / Withdrawal to painful stimuli** = 4<br>**Localizes to painful stimuli** = 5<br>**Obeys commands** = 6 |
| Glasgow coma scale verbal response | **Not Testable** = 0<br>**Makes no sounds** = 1<br>**Makes sounds** = 2<br>**Words** = 3<br>**Confused, disoriented** = 4<br>**Oriented, converses normally** = 5 |

PubMed [19]. The second part is using this data to train a word embedding model, this consists of selecting a model and setting hyperparameters. Both these steps are described below.

### 3.2.1 Gathering data from PubMed

PubMed is a database from the U.S. National Library of Medicine and the National Center for Biotechnology Information [19]. It contains more than 32 million citations and abstracts of biomedical literature. The database does not have full texts but may have links to PubMed Central and other websites, where full-text content can be obtained. So an important thing to note here is that all the texts we use to collect data from are the abstracts. However, these often do a good job at presenting the conclusion and findings of articles, which should make them work for the purpose of this work.

To access the data from PubMed, the PubTator Central (PTC) API is used [36]. This is a web-based automatic annotation system that interacts with both PubMed abstracts and their own full-text articles. Here their API is used for the purpose of downloading the raw data from the PubMed abstracts, this can be found in the Python script 'download-data-pubtator-search.py' in the GitHub repository linked in Section 3. Due to the limitations of our machines which will process the data downloaded and train the embedding model we have to restrict us regarding what abstracts to download. For each of the variables, from Table 3.3 (also described in Section 3.1.2.1), we come up with some set of keywords that are queried to PubMed to get the relevant articles. These keywords come from the medical terms of abnormally high or low values for each of the variables. PubMed

also has an automatic tag-system in order to find articles related to the topics queried, which means we should expect to get pretty good results despite not having any medical expertise. The keywords used in the queries can be viewed in Table 3.6. As an example of a query we will look at "Diastolic blood pressure", its search terms are used to construct the following PubMed query: `https://pubmed.ncbi.nlm.nih.gov/?term=high+diastolic+blood+pressure+OR+diastolic+blood+pressure+OR+low+diastolic+blood+pressure+OR+hypertension+OR+hypotension+OR+elevated+blood+pressure&filter=age.alladult`. The filter tag is used since if it was removed the query would reutrn over a million results. This query contains all the necessary information to fetch the abstracts data through the API. For each abstract received through they are saved as a new line in a per-variable document to be used later to construct the final corpus. Along with the texts themselves, their IDs are also saved, this identifier is unique per abstract and will be used to filter out duplicates when combining all texts into a larger file. The reason for duplicates is that some articles might relate to topics from multiple variables, e.g. both blood pressure and heart rate, and will thus appear in multiple queries.

The amount of abstracts found per variable varies a lot, some contain over a million relevant matches while others contain only a few thousand. In order to limit the data to be of a manageable scale, we download a maximum of 500 000 abstracts per variable. Regarding exactly which abstracts are chosen, the articles queried from PubMed are sorted according to their date of publication, meaning that they will be downloaded in that order. Some abstracts in PubMed are tagged with an age group which we sometimes use to find texts that are based only on adult physiology, 19 plus is the filter used here. This is however only used if the query still returns over 500 000 results. The reason for not always filtering is that a lot of reports are simply not tagged with an age group. The exact reasons as to why that is are unclear, but a vast majority of abstracts are still based on studies on adults which should make the data gathered relevant either way.

### 3.2.2 Word embedding model and creating a corpus

Before the word embedding model is trained there is some further preprocessing of the PubMed data that has to be done. Firstly a corpus needs to be constructed, a corpus is a large collection of words, which is needed in order to train a word embedding model. When the abstracts for a variable are downloaded, as described in the previous section, they are all stored in their own individual file, along with a file listing the IDs of all abstracts downloaded. To create the corpus all these abstracts need to be combined into a single file. This is done manually, for both the abstracts and all the ID lists. Using the list of IDs duplicates are then detected and those abstracts are then removed.

Now the preprocessing of the relevant data itself can begin. It should be noted that all parts from here on make use of an already existing library. We use Gensims (https://radimrehurek.com/gensim/models/word2vec.html) library whose im-

plementation of Word2Vec which is based of the original [20] [5]. Along with the Word2Vec implementation itself, the library also comes with a lot of functions to preprocess data for corpuses. Here we use the function called "simple_preprocess" which does a variety of actions, it removes any words shorter than 2 characters, or longer than 15, it converts all characters into lowercase, and lastly removes any character which is not a letter. It would also be possible to remove words that occur less frequently in this step, which is often done to eliminate nonsense or misspellings, but instead this is done when training the model. Now the corpus is ready to be used for training a Word2Vec model. The word embedding model is trained using the Continuous Bag Of Words (CBOW) algorithm, described in Section 2.7.1. The main reason for this over Skip-Gram is due to the faster speed of training, for future work it could perhaps be of interest to try the other algorithm. For the parameters used, it is trained for 20 epochs, meaning the corpus is iterated over 20 times, the vector size of the embeddings is 100, the minimum word count is set to 5, and the window size is set to 10.

## 3.3 Predicting in-hospital mortality

The prediction task chosen for the project was in-hospital mortality of patients. This was deemed to be a good starting point since it did not require much if any expertise within the medical field to get started with. For this task, the data from the first 48 hours of a patient's ICU stay is used to make a prediction of whether or not they will decease during their stay. The goal of such a prediction task is that it hopefully can be used to assist medical workers in alerting about patients whose health might be in critical condition.

### 3.3.1 Numerical features

Creating a good feature set was a challenging task. Each time series contained varying amounts of data, with many values missing due to measurements being made irregularly depending on what they measured. For example, heart rate might be measured more often compared to glucose levels. When exploring the design of the features we came up with some important things that need to be considered. The features need to capture some of the essential information that a time series provides, such as the development and relative changes between measured values, as well as changes developing over time. Early attempts did a poor job at this. For example, an early test was made where the features consisted of summaries of each column in the time series. It was figured that this removed all possibilities for a model to identify changes over time for a patient, such as if their health quickly deteriorates or recovers.

Further into the project another approach was tested where the time series is split into slices and information extracted from each of those. This idea comes from the MIMIC-III benchmark paper [9], which in turn was inspired by another research paper [37]. What we do is look at different subsets of the time series and extract some

statistical measurements from each. These subsets consist of the whole time series as well as the first and last 10/25/50 percent. For each subset, the minimum value, maximum value, mean value, standard deviation, skewness, and amount of measured values are calculated. This is repeated for each variable tracked in the time series. What this means is that for each of the 7 subsets there are 6 statistical observations made for each of the 16 variables tracked. The features are then standardised, this is common machine learning practice since many estimators perform poorly if the data is not normally distributed, this is done by removing the mean and scaling to unit variance. This results in feature vectors of size [1 x 672]. Overall this feature design we felt did a much better job preserving and capturing the details and development of measured values over time.

### 3.3.2 Categorical features

The categorical features are extracted in a similar fashion to the numerical features described in Section 3.3.1. The only difference is that they are calculated using the discrete categorical values instead of the values directly from a patient's measurement. This process is described in Section 3.1.4.

### 3.3.3 Biomedical features

The biomedical features are added as a concatenation to the already existing features. They are added to both the numerical and categorical features in order to train models on these for comparison. These features are derived from the word embedding model trained on the PubMed articles, described in Section 3.2. This was a big challenge since it was not immediately obvious how to create these. Each categorised value from a patient's time series could be turned into a word vector which leaves many possibilities on how to create these features. To generate the biomedical features of an episode we first look at each variable column and translate the categorised values into their respective medical term, such as "hypertension" or "hypotension". If there exists an empty value these are just removed.

There are however some values that do not categorise into specific medical terms that are omitted when it comes to extracting word vectors. Examples of these are the Glasgow coma scales and capillary refill rate. Since the word embeddings model used uses the standard Word2Vec implementation it only supports unigrams, of which the Glasgow scale does not have any. Capillary refill rate is categorised into *normal* and *abnormal*, and lacks any specific medical terms which also meant that no good word vectors could be found. Since no good vector representations could be found these are ignored when creating the biomedical features.

For each variable in an episode, all corresponding word vectors are weighted according to how many values were measured. Then all values, for all variables are summarised into a single word vector that represents the whole episode. These are then added as a concatenation for both the numerical and categorical features, resulting in a feature vector of size [1 x 762].

### 3.3.4 Training the prediction models

Three algorithms that will be used to train different prediction models are Logistic regression, Random forest, and SVM, which are all described in Section 2.3. Each algorithm will be used to train a model on each of the feature sets described in Sections 3.3.1-3.3.3, meaning a total of 12 models to evaluate. Then each numerical and categorical model will be compared to its biomedical counterpart, this is done per algorithm used. The performance will be asserted using a stratified K-fold cross-validation described in Section 2.4.1. The exact details regarding how each individual model is trained and evaluated are all presented along with their respective results in Section 4.1.

**Table 3.6:** Search terms queried to PubMed for each variable.

| **Variable** | **Search terms** |
|---|---|
| Capillary refill rate | capillary refill rate, crt, capillary refill time |
| Diastolic blood pressure | high diastolic blood pressure, diastolic blood pressure, low diastolic blood pressure, hypertension, hypotension, elevated blood pressure |
| Fraction inspired oxygen | fraction inspired oxygen, fio2, hyperoxia, hypoxia |
| Glasgow coma scale eye opening | glasgow coma scale eye opening, gcs eye opening, glasgow coma scale, gcs |
| Glasgow coma scale motor response | glasgow coma scale motor response, gcs motor response, glasgow coma scale, gcs |
| Glascow coma scale verbal response | glascow coma scale verbal response, gcs verbal respons, glascow coma scale, gcs |
| Glucose | glucose, blood sugar, low blood sugar, high blood sugar, hyperglycemia, hypoglycemia |
| Frequency heart rate | frequency heart rate, high frequency heart rate, low frequency heart rate, tachycardia, bradycardia |
| Height | height, short stature, tall stature |
| Mean blood pressure | mean blood pressure, elevated blood pressure, hypertension, hypotension |
| Oxygen saturation | oxygen saturation, hypoxemia, hypoxia |
| Respiratory rate | respiratory rate, tachypnea, bradypnea, eupnoea |
| Systolic blood pressure | high systolic blood pressure, systolic blood pressure, low systolic blood pressure, hypertension, hypotension, elevated blood pressure |
| Temperature | body temperature, low body temperature, high body temperature, hypothermia, fever, hyperpyrexia, hyperthermia |
| Weight | weight, body weight, underweight, overweight, obesity |
| pH | "blood ph levels", blood ph acidosis, blood ph alkalosis, acidemia, acidosis |

# 4
# Results

## 4.1 In-hospital mortality prediction

To evaluate the project a number of models were trained and evaluated using some of the mentioned metrics in Section 2.5. These are accuracy, precision, recall, F1 score, ROC curve, and PR curve. In order to be able to study what impact the different features have on the results, three different algorithms were chosen for the models, these are described in Section 2.3. For each algorithm the performance difference with and without the biomedical features is measured.

The data set contains 4487 positive cases and the feature vectors used are of sizes [1 x 672] and [1 x 762]. From our testing we found that the results of a model varied greatly depending on how the positive cases were split among the training and test data. To handle this each model is trained using a 10 fold cross-validation, described in Section 2.4.1, and their results are averaged and presented below.

### 4.1.1 Logistic regression models

The results of the logistic regression models can be seen in Tables 4.1, 4.2, and Figures 4.1-4.4. What can be seen here in the extended features from the word embeddings make very little difference in the results. The accuracy remains the same for both the numerical and categorical models. The precision varies a bit, it is a bit worse for the numerical and a bit better for the categorical. The recall sees an improvement of a few percent for both types and thus the F1 score also improves slightly.

As a consequence of the small change in the previously presented results the ROC and PR curves also show very little difference when adding the biomedical features.

The logistic regression model uses L2 regularisation with C=1 and Broyden-Fletcher-Goldfarb-Shanno (LBFGS) as the solver. The same setup is used for both the numerical and categorical models.

**Table 4.1:** Results of the numerical logistic regression models

| | **Numerical** | **Numerical + biomedical** | **Percentage difference** |
|---|---|---|---|
| **Accuracy** | 0.876 | 0.876 | |
| **Std dev** | 0.003 | 0.003 | |
| **Precision** | 0.615 | 0.614 | -0.2% |
| **Std dev** | 0.027 | 0.026 | |
| **Recall** | 0.272 | 0.284 | 4% |
| **Std dev** | 0.019 | 0.020 | |
| **F1_score** | 0.377 | 0.388 | 3% |
| **Std dev** | 0.021 | 0.022 | |

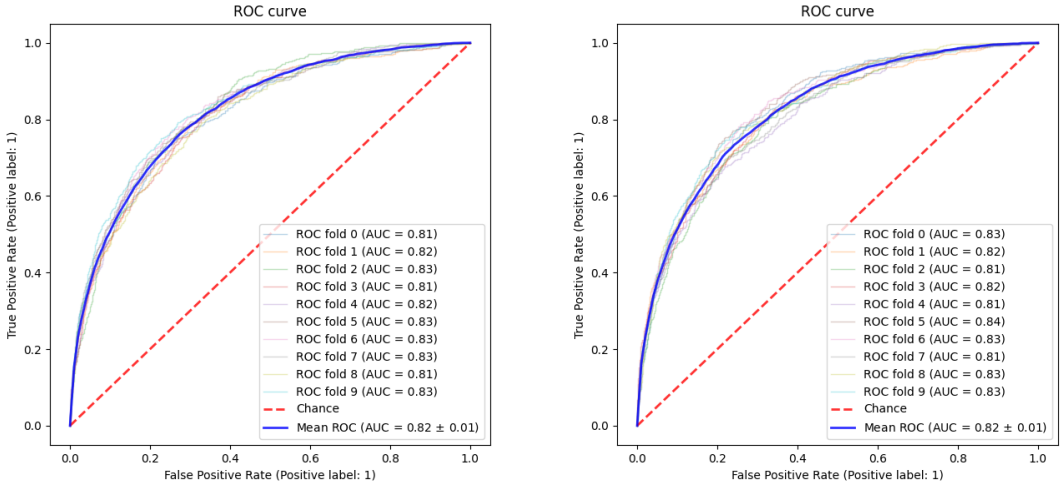


**Figure 4.1:** Both images show the logistic regression, trained using 10 fold cross-validation. The left image is with numerical features and the right image is with numerical and biomedical features.

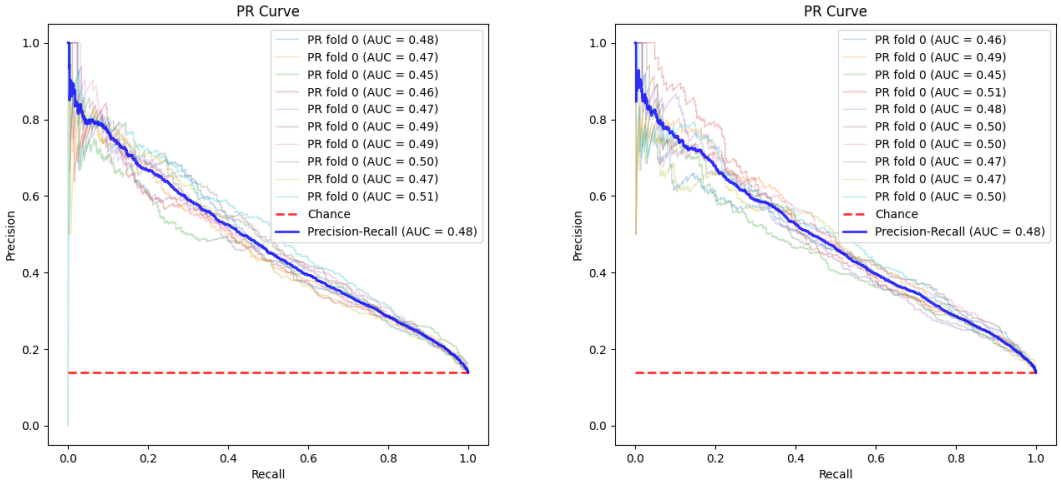


**Figure 4.2:** Both images show the logistic regression, trained using 10 fold cross-validation. The left image is with numerical features and the right image is with numerical and biomedical features.

**Table 4.2:** Results of the categorical logistic regression models

| | **Categorical** | **Categorical + biomedical** | **Percentage difference** |
|---|---|---|---|
| **Accuracy** | 0.871 | 0.872 | |
| **Std dev** | 0.004 | 0.004 | |
| **Precision** | 0.590 | 0.596 | 1% |
| **Std dev** | 0.036 | 0.033 | |
| **Recall** | 0.238 | 0.245 | 3% |
| **Std dev** | 0.021 | 0.017 | |
| **F1_score** | 0.339 | 0.347 | 2% |
| **Std dev** | 0.026 | 0.020 | |

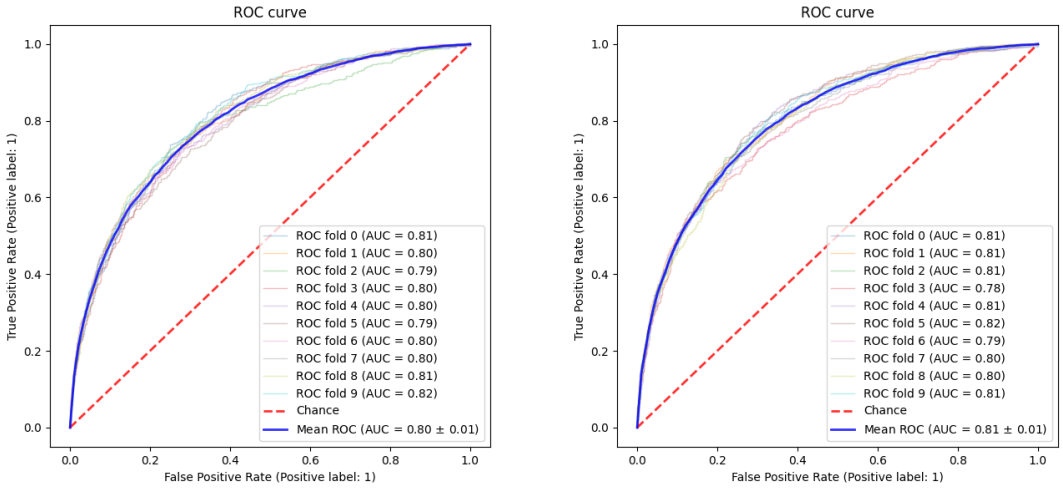


**Figure 4.3:** Both images show the logistic regression, trained using 10 fold cross-validation. The left image is with categorical features and the right image is with categorical and biomedical features.

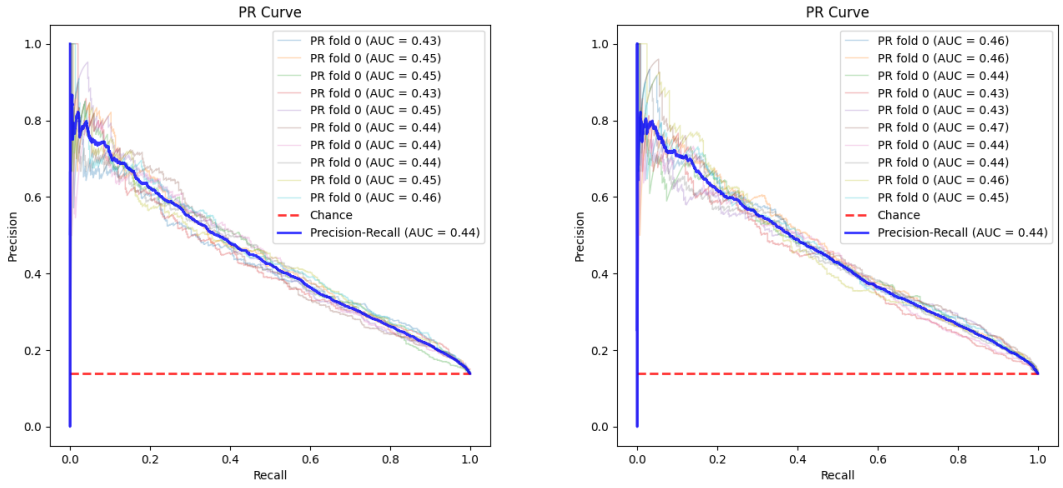


**Figure 4.4:** Both images show the logistic regression, trained using 10 fold cross-validation. The left image is with categorical features and the right image is with categorical and biomedical features.

### 4.1.2 Random forest models

The results of the random forest models can be seen in Tables 4.3, 4.4, and Figures 4.5-4.8. What can be seen here is a similar result to the logistic regression models. The accuracy remains almost the same, a small change can be seen in the precision, in this case both are negative, and an improvement in recall, leading to a higher F1 score.

As seen with the logistic regression models the small changes in the accuracy, precision, and recall are reflected in the ROC and PR curves and they remain almost the same with the addition of the biomedical features.

The random forest model is trained using 100 estimators, both for the numerical and categorical models.

**Table 4.3:** Results of the numerical random forest models

| | Numerical | Numerical + biomedical | Percentage difference |
|---|---|---|---|
| **Accuracy** | 0.877 | 0.878 | |
| **Std dev** | 0.003 | 0.002 | |
| **Precision** | 0.713 | 0.708 | -1% |
| **Std dev** | 0.047 | 0.019 | |
| **Recall** | 0.191 | 0.200 | 5% |
| **Std dev** | 0.021 | 0.016 | |
| **F1_score** | 0.300 | 0.312 | 4% |
| **Std dev** | 0.026 | 0.020 | |

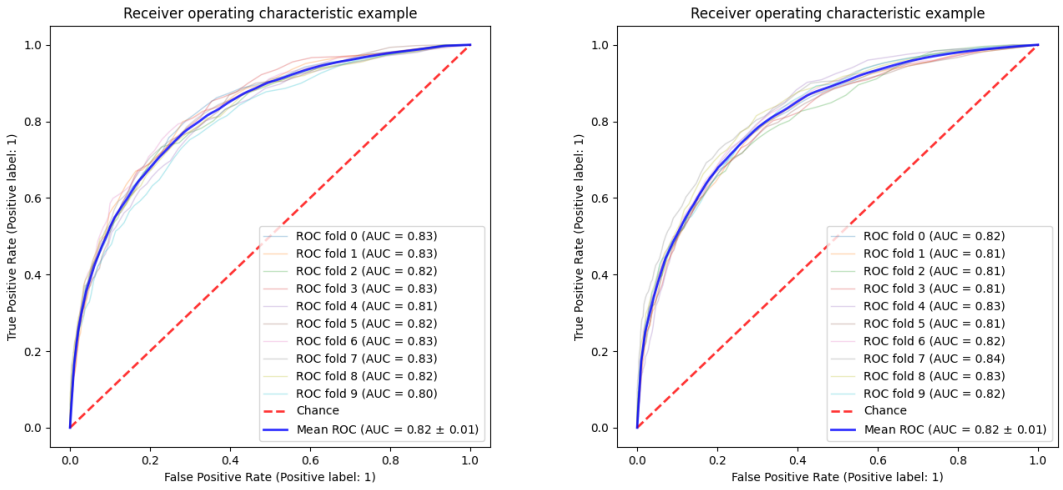


**Figure 4.5:** Both images show the random forest, trained using 10 fold cross-validation. The left image is with numerical features and the right image is with numerical and biomedical features.

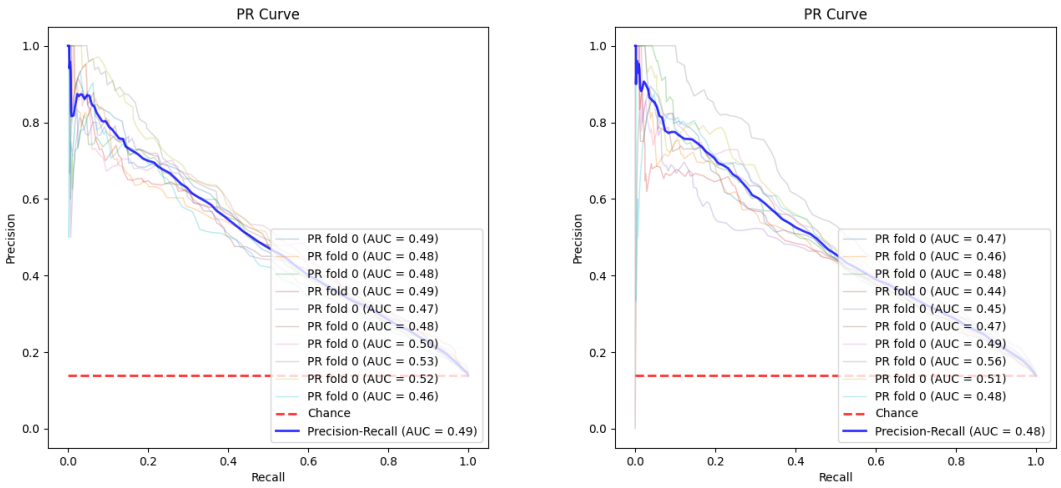


**Figure 4.6:** Both images show the random forest, trained using 10 fold cross-validation. The left image is with numerical features and the right image is with numerical and biomedical features.

**Table 4.4:** Results of the categorical random forest models

| | **Categorical** | **Categorical + biomedical** | **Percentage difference** |
|---|---|---|---|
| **Accuracy** | 0.874 | 0.874 | |
| **Std dev** | 0.003 | 0.002 | |
| **Precision** | 0.680 | 0.669 | -2% |
| **Std dev** | 0.055 | 0.029 | |
| **Recall** | 0.167 | 0.186 | 11% |
| **Std dev** | 0.014 | 0.013 | |
| **F1_score** | 0.268 | 0.291 | 9% |
| **Std dev** | 0.021 | 0.017 | |

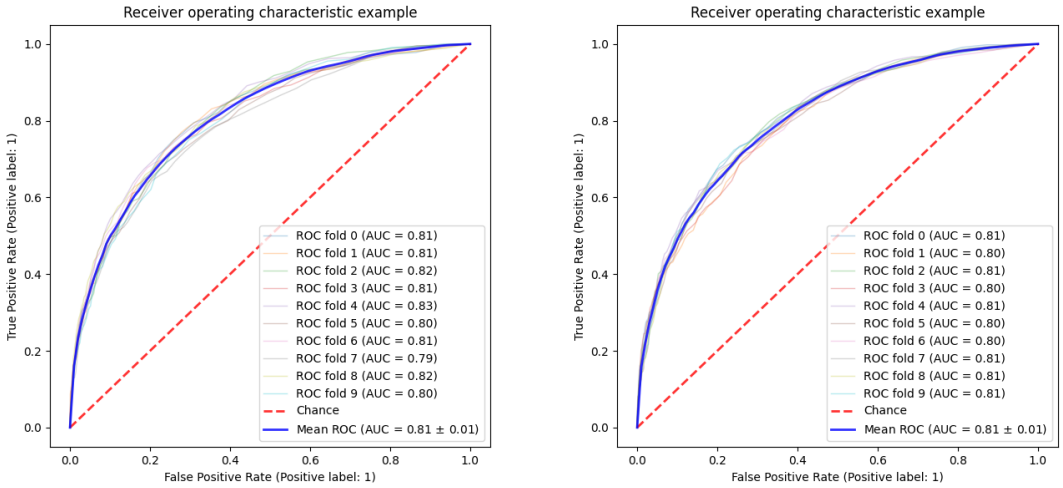


**Figure 4.7:** Both images show the random forest, trained using 10 fold cross-validation. The left image is with categorical features and the right image is with categorical and biomedical features.

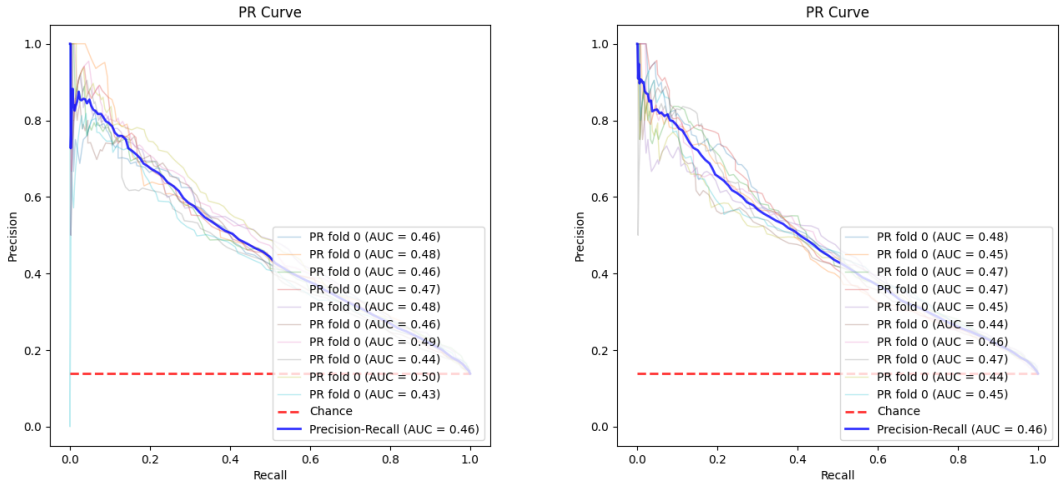


**Figure 4.8:** Both images show the random forest, trained using 10 fold cross-validation. The left image is with categorical features and the right image is with categorical and biomedical features.

### 4.1.3 Support vector classifier model

The results of the support vector classifier models can be seen in Tables 4.5, 4.6, and Figures 4.9-4.12. The results are very similar to the ones that can be seen for the previous two algorithms. The accuracy remains barely changed, a small difference in precision is seen as well as an improvement in recall and F1 score.

Since the accuracy, precision, and recall follow the same trend as seen for the previous models we once again see little difference in the ROC and PR curves.

This is using default hyperparameters of the SVC implementation in Scikit-learn, which uses Radial Basis Function (RBF) as the kernel.

**Table 4.5:** Results of the numerical SVC models

| | **Numerical** | **Numerical + biomedical** | **Percentage difference** |
|---|---|---|---|
| **Accuracy** | 0.876 | 0.877 | |
| **Std dev** | 0.002 | 0.003 | |
| **Precision** | 0.731 | 0.719 | -2% |
| **Std dev** | 0.027 | 0.030 | |
| **Recall** | 0.168 | 0.181 | 8% |
| **Std dev** | 0.013 | 0.017 | |
| **F1_score** | 0.273 | 0.289 | 6% |
| **Std dev** | 0.017 | 0.023 | |

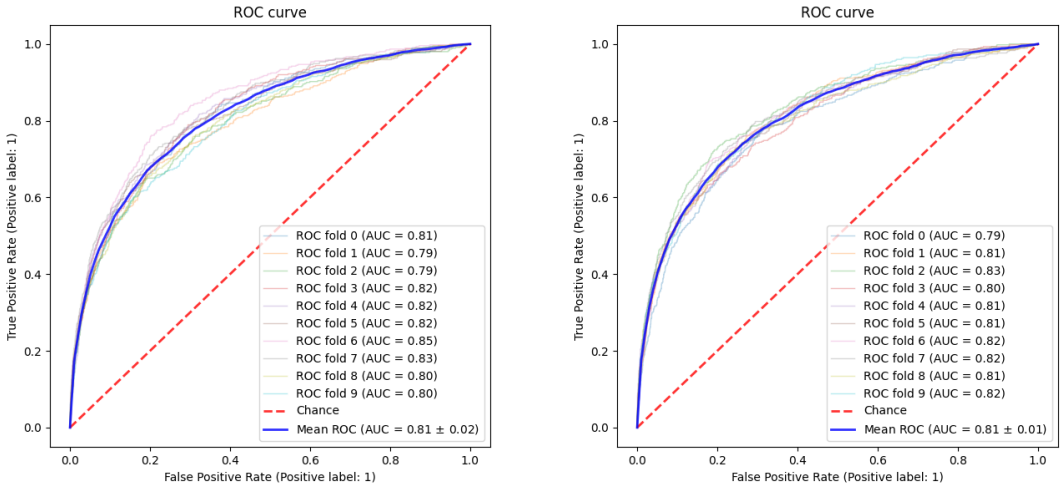


**Figure 4.9:** Both images show the SVC, trained using 10 fold cross-validation. The left image is with numerical features and the right image is with numerical and biomedical features.

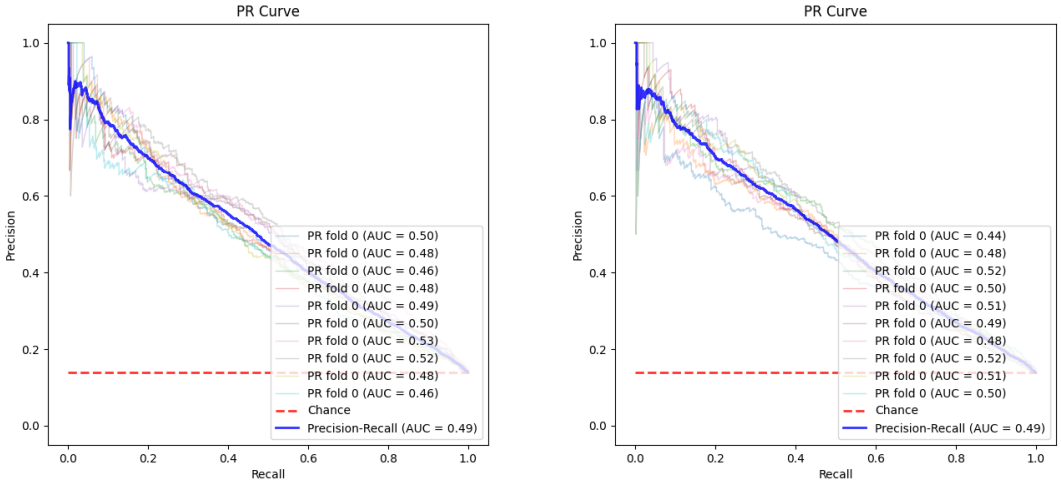


**Figure 4.10:** Both images show the SVC, trained using 10 fold cross-validation. The left image is with numerical features and the right image is with numerical and biomedical features.

**Table 4.6:** Results of the categorical SVC models

| | Cateogrical | Categorical + biomedical | Percentage difference |
|---|---|---|---|
| **Accuracy** | 0.873 | 0.873 | |
| **Std dev** | 0.003 | 0.002 | |
| **Precision** | 0.700 | 0.694 | -1% |
| **Std dev** | 0.048 | 0.032 | |
| **Recall** | 0.140 | 0.148 | 6% |
| **Std dev** | 0.019 | 0.009 | |
| **F1_score** | 0.233 | 0.243 | 4% |
| **Std dev** | 0.028 | 0.013 | |

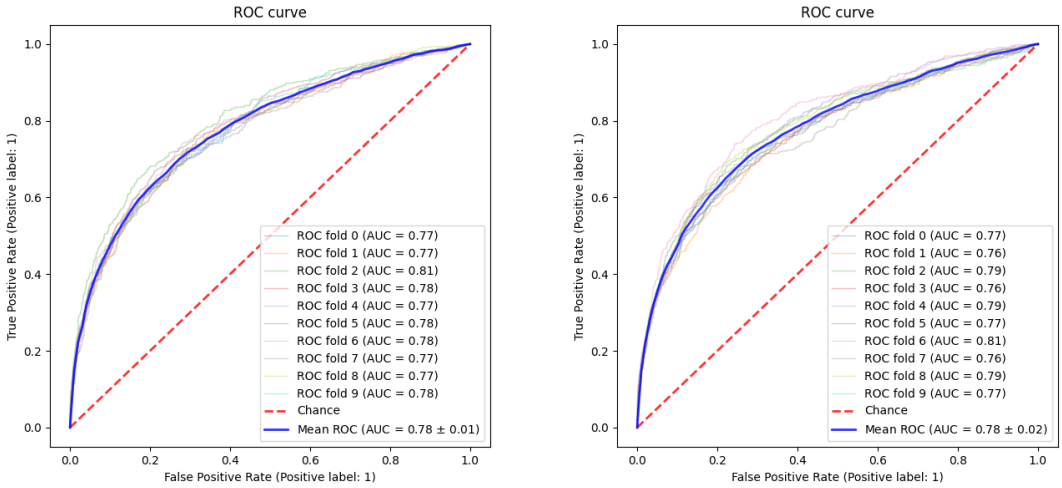


**Figure 4.11:** Both images show the SVC, trained using 10 fold cross-validation.The left image is with categorical features and the right image is with categorical and biomedical features.

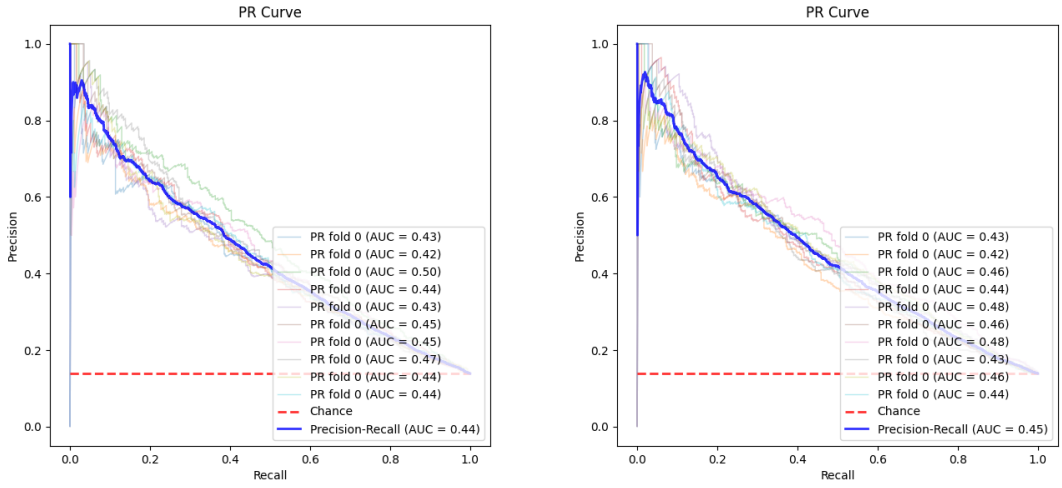


**Figure 4.12:** Both images show the SVC, trained using 10 fold cross-validation. The left image is with categorical features and the right image is with categorical and biomedical features.

# 5
# Conclusion

This section contains a discussion of ideas on how this project could progress further as well as the conclusion that we draw from the results obtained.

## 5.1 Discussion

Due to the time frame of the project and the fact that some things took longer than initially expected, there were a lot of ideas that were thought could improve the project but that could not be implemented. Most of these issues come from the fact that the process of going from raw data to a first numerical prediction model took longer time than expected. We thought that the benchmark paper would require quite little work in order to get a baseline prediction model. Thus allowing us to put more effort into training the word embeddings and creating the biomedical features. But since the benchmark project had not been maintained for a while it required some work in order for it to run. This required us to modify their code, but we were unsure if this was a good idea. If we wanted to modify it we felt that we needed to completely understand it in to ensure it was working correctly, but this was a large project that contained quite a lot of code. This in combination with MIMIC-IV being released led us to change our original plans. Instead of spending all the time on understanding the code of the benchmark project we wrote our own program which did only what we needed, with the old paper and project as a starting point while using the newer database instead. Unfortunately, this led to less time allocated to working on the biomedical features.

### 5.1.1 Word embeddings and the information we extract from them

The main question we asked ourselves at the beginning of the report is if we can extract information from scientific biomedical literature which can be used to improve patient outcome prediction models. This can be done in a plethora of ways, eg relating different symptoms/diseases to each other, making some sort of "network" of how all these connect. Depending on what has desired another type of embedding could be used, for example, a transformer model such as BERT [38]. Due to the time constraints within this project this was not explored, but its usefulness could be evaluated for future work on further incorporating scientific literature into these types of models.

### 5.1.2 Features and the amount of data available

A problem that was discussed quite a few times during the project was how to design our features, here we discuss the numerical and categorical ones. The main problem faced was that each individual patient had a different amount of measured lab results. This made it difficult to come up with a feature set that could be extracted for each patient. We eventually settled on an approach that was introduced in a different paper, this is discussed in Section 3.3.1. This however resulted in a substantially large amount of features, which we initially did not think too much about. We do however suspect that this might lead to high variance for the models, meaning a higher risk of overfitting. This could be seen in the high variance in the performance metrics when using different data splits. For the majority of the project we used a static split, meaning every fit was done on the same split with 85% of the data as the training set and 15% as the test set. However with 4487 positive cases, 27 906 negative cases, and 672 features for the numerical and categorical models we suspect that this might have been too many features. If this is an issue then it could become even worse when adding the additional 100 features from the word embeddings. Thus their addition might give the model more variance and their potential benefit might not be as prevalent. Cross-validation should help mitigate some of this, but more data or fewer features might still be preferable. We also discussed the idea of tracking more variables/lab results. But following the same reasoning as above we decided to stick with the initial ones in fear of overfitting the model with too many features.

### 5.1.3 Choice of prediction tasks

Due to the time limit of the project, the focus was put on creating a single prediction task, predicting the mortality risk of a patient based on their first 48 hours in ICU. This was chosen for a variety of reasons, both we and our supervisor thought it was a good starting point for the project. This does however pose some points of discussion regarding the conclusions that can be drawn from the results. While we still draw the conclusion that word embeddings could be useful for these types of prediction tasks, more prediction tasks could be implemented to confirm it. For example some of the tasks mentioned in Section 2.2, some of these might be more suitable to benefit from the word embeddings.

## 5.2 Conclusion

Our results could point towards there being valuable information that could be extracted from biomedical texts through word embeddings, which could then be used in strengthening prediction models. Unfortunately, we deem the differences in performance to be too small to conclude that they do indeed offer an improvement, but we do however believe it suggest that further exploration should be made on the topic, such as exploring different methods of word embeddings or revisiting the feature engineering.

# Bibliography


[1] T. Yiu, "Understanding random forest," 2019, last accessed on 2021-11-28. [Online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[2] R. Gandhi, "Support vector machine — introduction to machine learning algorithms," 2018, last accessed on 2021-11-28. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[3] J. Brownlee, "How to use roc curves and precision-recall curves for classification in python," 2018, last accessed on 2021-11-24. [Online]. Available: https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/

[4] M. Galarnyk, "Understanding boxplots," 2018, last accessed on 2021-11-24. [Online]. Available: https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013.

[6] P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, M. P. Lungren, and A. Y. Ng, "Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning," 2017.

[7] A. McCoy and R. Das, "Reducing patient mortality, length of stay and readmissions through machine learning-based sepsis prediction in the emergency department, intensive care unit and hospital floor units," *BMJ Open Quality*, vol. 6, no. 2, 2017. [Online]. Available: https://bmjopenquality.bmj.com/content/6/2/e000158

[8] A. Johnson, L. Bulgarelli, T. Pollard, S. Horng, L. A. Celi, and R. Mark, "Mimic_iv (version 1.0)," 2021.

[9] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. Ver Steeg, and A. Galstyan, "Multitask learning and benchmarking with clinical time series data," *Scientific Data*, vol. 6, no. 1, p. 96, 2019. [Online]. Available: https://doi.org/10.1038/s41597-019-0103-9

[10] T. Yiu, "Understanding logistic regression," 2019, last accessed on 2021-11-28. [Online]. Available: https://towardsdatascience.com/understanding-logistic-regression-using-a-simple-example-163de52ea900

[11] O. Mbaabu, "Introduction to random forest in machine learning," 2020, last accessed on 2021-11-28. [Online]. Available: https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/

[12] D. Berrar, "Cross-validation." 2019.

[13] P. Refaeilzadeh, L. Tang, and H. Liu, *Cross-Validation*. New York, NY: Springer New York, 2016, pp. 1–7. [Online]. Available: https://doi.org/10.1007/978-1-4899-7993-3_565-2

[14] M. Bekkar, H. K. Djemaa, and T. A. Alitouche, "Evaluation measures for models assessment over imbalanced data sets," *J Inf Eng Appl*, vol. 3, no. 10, 2013.

[15] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[16] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets," *PLOS ONE*, vol. 10, no. 3, pp. 1–21, 03 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0118432

[17] B. Jason, "How to remove outliers for machine learning," 2018, last accessed on 2021-06-03. [Online]. Available: https://machinelearningmastery.com/how-to-use-statistics-to-identify-outliers-in-data/?fbclid=IwAR1EquiAreYrWnjXhK1G3gsy4bfnl7JyCoGJa2Bt-dflX3xeTvc_8CzH2eM

[18] G. Michael, "Understanding boxplots," 2018, last accessed on 2021-06-03. [Online]. Available: https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51

[19] "Pubmed," last accessed on 2021-05-28. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[21] "Anthropometric reference data for children and adults: United states, 2003–2006." [Online]. Available: https://www.cdc.gov/nchs/data/nhsr/nhsr010.pdf

[22] S. Pedicelli, E. Peschiaroli, E. Violi, and S. Cianfarani, "Controversies in the definition and treatment of idiopathic short stature (iss)," 2009. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3005647/

[23] S. Kumar, "Tall stature in children: differential diagnosis and management," 2013.

[24] "Högt blodtryck," last accessed on 2021-05-20. [Online]. Available: https://www.hjart-lungfonden.se/halsa/riskfaktorer/hogt-blodtryck/

[25] "Lågt blodtryck," last accessed on 2021-05-20. [Online]. Available: https://www.1177.se/Vastra-Gotaland/sjukdomar--besvar/hjarta-och-blodkarl/blodtryck/lagt-blodtryck/

[26] G. Teasdale, "Glasgow coma scale: Do it this way," 2015. [Online]. Available: https://www.glasgowcomascale.org/downloads/GCS-Assessment-Aid-English.pdf?v=3

[27] P. Jevon and H. Gallier, "How to measure capillary refill time in patients who are acutely ill," vol. 116, no. 8, pp. 29–30, 2020. [Online]. Available: https://www.nursingtimes.net/clinical-archive/assessment-skills/how-to-measure-capillary-refill-time-in-patients-who-are-acutely-ill-20-07-2020/

[28] S. Fuentes and Y. S. Chowdhury, "Fraction of inspired oxygen," Jan 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK560867/

[29] D. Castro, S. M. Patil, and M. Keenaghan, "Arterial blood gas," Jan 2021, last accessed on 2021-05-20. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK536919/

[30] "Vital signs (body temperature, pulse rate, respiration rate, blood pressure)," last accessed on 2021-05-20. [Online]. Available: https://www.hopkinsmedicine.org/health/conditions-and-diseases/vital-signs-body-temperature-pulse-rate-respiration-rate-blood-pressure

[31] "Blodprov: P-glukos - blodsocker," last accessed on 2021-05-20. [Online]. Available: https://www.1177.se/Vastra-Gotaland/behandling--hjalpmedel/undersokningar-och-provtagning/provtagning-och-matningar/blodprov/blodprov-p-glukos---blodsocker/

[32] "Feber," last accessed on 2021-05-20. [Online]. Available: https://www.1177.se/Vastra-Gotaland/sjukdomar--besvar/infektioner/feber/feber/

[33] "Diabetes and hyperglycemia," last accessed on 2021-09-23. [Online]. Available: https://www.diabetes.co.uk/Diabetes-and-Hyperglycaemia.html

[34] "Pulspalpation, pulsmätning - Översikt," last accessed on 2021-05-20. [Online]. Available: https://www.vardhandboken.se/undersokning-och-provtagning/pulspalpation-pulsmatning/oversikt/

[35] "Oxygenbehandling - Översikt," last accessed on 2021-05-20. [Online]. Available: https://www.vardhandboken.se/vard-och-behandling/lakemedelsbehandling/oxygenbehandling/oversikt/

[36] "Pubtator central," last accessed on 2021-11-16. [Online]. Available: https://www.ncbi.nlm.nih.gov/research/pubtator/

[37] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzel, "Learning to diagnose with lstm recurrent neural networks," 2017.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.