FINAL DEGREE PROJECT

Degree in Biomedical Engineering

PREDICTION OF EXTRA-HOSPITAL MORTALITY IN CRITICAL PATIENTS THROUGH NEURAL NETWORKS ARTIFICIAL

Report and Annexes

Author: Daniel Solá Fraire
Director: Samir Kanaan Izquierdo
Announcement: October 2018

index

Summary

According to the WHO, a medical destination is a change in the health status of an individual, group, or population, attributable to specific interventions. Prediction of medical destinations is a branch of medical research that studies the final result of the structure and processes of the medical system on the health and well-being of patients and populations. It is an important pillar in decision-making and in the analysis of policies and procedures, since it allows the evaluation of the quality of medical care, its efficiency and effectiveness.
Its objective is to identify failures in medical practice that affect the patient's health and develop strategies that improve care. To do this, it measures tangible events experienced by the patient, such as mortality, readmission or morbidity. The result of the research on medical destinations is used to inform legislative bodies that make decisions related to health, as well as financial bodies, such as the government or insurance companies, that seek to minimize medical costs by providing adequate medical care. The standardized collection of statistics and medical data about the medical care that patients receive has allowed medical records to be used as a reliable source for research.

One of the most relevant medical destinations is the out-of-hospital mortality of patients, that is, the time until death after hospital discharge. The objective of this study is the prediction of these variables using artificial neural networks. Specifically, it is a classifi-catory prediction task in three time frames (0 - 1 months, 1 - 12 months, 12+ months). For this purpose, a total of 43 predictive variables are used, including demographic information, physiological signs, laboratory test results and other variables related to the patients' hospital stay. We obtain the necessary information from the public database MIMIC-III v1.4, corresponding to hospital admissions to the intensive care unit at the Beth Israel Deaconess Medical Center hospital, in Boston, Massachusetts, USA. This is a data set that contains deidentified medical information on more than 40,000 critically ill patients between the years 2001 and 2012. In this way, we will design a neural network capable of successfully predict out-of-hospital mortality of patients based on these variables.

Keywords—Artificial Neural Networks, data mining, artificial intelligence, mortality prediction, deep learning, intensive care unit

Abstract

According to WHO, a medical outcome is a change in the health status of an individual, group, or population, attributable to certain causes. The prediction of medical outcomes is a branch of medical research that studies the final result of the structure and processes of the medical system in the health and well-being of patients and populations. It is an important factor in decision making and analysis of policies and procedures, since it allows the evaluation of the quality of medical care, its efficiency and effectiveness. Its objective is to identify faults in medical practice that affect the health of the patient and the development of strategies that improve care. To do this, it measures tangible events experienced by the patient, stories such as mortality, readmission or morbidity. The result of the research on medical destinations is used to inform the legal bodies that make decisions related to health, as well as financial entities, such as the government or insurance companies, that seek to minimize costs while providing adequate medical care. Routine collection statistics and medical data related to patient care that patients have allowed medical records to be used as a reliable source in research.

One of the most relevant medical outcomes is out-of-hospital mortality of patients, that is, the time until their death after hospital discharge. The objective of this study is the prediction of this outcome through artificial neural networks. Specifically, it is a classificatory prediction task in three time intervals (0 - 1 months, 1 - 12 months, 12+ months). For this, a total of 43 predictor variables are used, including demographic information, physiological signals, results of laboratory tests and other variables related to the hospital stay of patients. Data is obtained from the public database MIMIC-III v1.4, corresponding to hospital admissions in the intensive care unit at Beth Israel Deaconess Medical Center, in Boston, Massachusetts, USA. It is a set of data that contains medical information of more than 40,000 critical patients between 2001 and 2012. In this way, we will design an artificial neural network capable of satisfactorily predicting out-of-hospital mortality of patients based on these variables.

Keywords— Artificial Neural Networks, Data mining, Artificial Intelligence, Mortality prediction, Deep Learning, Intensive Care Unit

1. Introduction

Artificial intelligence is a relatively recent field with multiple applications in various areas, including medicine. It is the ability of computer algorithms to approximate conclusions without human intervention. The main objective of the application of artificial intelligence to medicine is to analyze the relationships between prevention techniques or treatments and their results on patients. Currently, solutions have been developed for various problems in diagnostic processes, drug development, personalized medicine, patient treatment and monitoring, etc. Large companies such as IBM and Google have also developed artificial intelligence algorithms for the healthcare sector. It is an expanding field, with constant research and great promises for the future.

One of the main applications of artificial intelligence in medicine is the prediction of medical destinations. According to the WHO, a medical destination is a change in the health status of an individual, group, or population, attributable to specific interventions. Prediction of medical destinations is a branch of medical research that studies the final result of the structure and processes of the medical system on the health and well-being of patients and populations.
It is an important pillar in decision-making and in the analysis of medical policies and procedures, since it allows the evaluation of the quality of the treatment, its efficiency and effectiveness. Its objective is to identify failures in medical practice that affect the patient's health and develop strategies that improve care. To do this, it measures tangible events experienced by the patient, such as mortality, readmission or morbidity.
The result of the research on medical destinations is used to inform legislative bodies that make decisions related to healthcare, as well as financial bodies, such as the government or insurance companies, that They seek to minimize medical costs by providing adequate medical care. The standardized collection of statistics and medical data about the medical care that patients receive has allowed medical records to be used as a reliable source for research.

One of the most relevant medical destinations is the out-of-hospital mortality of patients, that is, the time until death after hospital discharge. The objective of this study is the prediction of this variable using artificial neural networks. Specifically, it is a classificatory prediction task in three time frames (0 - 1 months, 1 - 12 months, 12+ months). For this purpose, a total of 43 predictive variables are used, including demographic information, physiological signs, laboratory test results and other variables related to the patients' hospital stay.

We obtain the necessary information from the public database MIMIC-III v1.4, corresponding to hospital admissions to the intensive care unit at the Beth Israel Deaconess Medical Center hospital, in Boston, Massachusetts, USA. It is a data set that contains identified medical information on more than 40,000 critically ill patients between the years 2001 and 2012. In this way, we will design a neural network capable of successfully predicting mortality. out-of-hospital status of patients based on these variables.

1.1 Objectives and scope

The main objective is the development and implementation of a classifying predictive model based on artificial neural networks capable of predicting out-of-hospital mortality in critically ill patients. It is desired that this model present an AUROC greater than 0.80, thus indicating good diagnostic capacity. Once the model is built, we want to analyze its performance on real data using different evaluation metrics, in order to draw relevant conclusions about its prediction capacity. Likewise, it will be compared with other similar predictive models to measure how it compares with the current state of the art and determine if an adequate result has been obtained.

The detailed interpretation of certain highly complex mathematical concepts, such as certain optimization algorithms, or the analysis of pathologies or medical situations and their effects on mortality, is outside the scope of this document. from the patients.

2. Description of the data set

MIMIC-III ("Medical Information Mart for Intensive Care") is a database corresponding to hospital admissions to the intensive care unit at the Beth Israel Deaconess Medical Center hospital, in Boston, Massachusetts, USA. It includes information relating to patients' vital signs, medications, laboratory measurements, observations and notes taken by medical personnel, fluid balance, procedure codes, diagnostic codes, immobility reports medical conditions, length of hospital stay and patient survival data, among others. Likewise, information about out-of-hospital mortality is collected from US social security files. These data are deidentified and are in the public domain for academic use. It is the only freely accessible database of its kind. Furthermore, it stands out for its large number of records, obtained over more than a decade, specifically between the years 2001 and 2012. It contains data associated with 53,423 hospital admissions for adult patients over 16 years of age. ages and information on 7870 neonates admitted between 2001 and 2008. The median age of the patients is 65.8 years, 55.9% of the patients are men, and in-hospital mortality is 11.5%. The median length of an intensive care unit stay is 2.1 days and the median length of stay in the hospital is 6.9 days.

Two devices were used as monitoring and data collection systems: Philips CareVue Clinical Information System (M2331A and M1215A) and iMDsoft MetaVision ICU. These devices were the source of various clinical data, such as physiological measures such as heart rate, blood pressure or respiratory rate, notes on patient progress or medication delivery. MIMIC - III merges the data coming from the two devices where possible.

2.1 Deidentification, privacy and access conditions

All data were deidentified before being entered into MIMIC-III, in accordance with current US regulations, "Health Insurance Portability and Accountability Act (HIPAA)". To do this, all information that allowed patients to be identified was eliminated, such as telephone number, name, address, etc.
As for the dates, they were moved in the future randomly in a consistent manner for each individual, resulting in stays that occurred between the year 2100 and 2200. However, the time, day of the week and season were preserved in this date modification process.
Likewise, the age of patients over 89 years of age was masked to preserve their privacy according to current regulations. That is why they appear at ages greater than 300 years.
To access the database, it is necessary to carry out a process consisting of completing a recognized course on data protection of study participants, in accordance with HIPPA regulations, and signing a use agreement, which delimits a appropriate use of information and security standards, in addition to expressly prohibiting user identification. The process requires about a week and is done online.
Specifically, you must take the 'Data or Specimens Only Research' course provided by the Massachusetts Institute of Technology through the CITI Program, Collaborative Institutional Training Initiative.

Figure 2.1: CITI program logo

Once completed, a certificate of completion is received, which must be sent to the MIMIC-III administrators in order to obtain the necessary access keys.

Once this process is done, the information is obtained as a collection of CSVs, along with scripts to import them into databases. On the official mimic website (https://mimic.physionet.org) you will find the steps and scripts necessary to load the files into a local PostgreSQL database, as well as to create indexes. search.

2.2 Tables

- ADMISSIONS: Defines the hospital admission of each patient, identifying each one with an ID, HADM ID. It contains 58976 records. The information comes from the hospital database. Likewise, it contains some entries related to the donation of organs from patients who died in the hospital.
- CALLOUT: This table contains information on patients ready to be discharged from the ICU. When this occurs, a patient is said to be 'Called out'. This information is not available for all patients, since it began to be collected after the database was created. Likewise, for unspecified reasons, entries relating to neonates are not included. It contains 34499 records.

  When a patient is ready to be discharged from the ICU, the medical staff in charge creates a 'call out' request, which is subsequently admitted. He is later transferred out of the ICU.
- CAREGIVERS: This table provides information about medical personnel and their interventions on patients. It contains 7567 records and comes from the database of the CareVue and Metavision monitoring devices.
- CHARTEVENTS: This table contains information relating to patients during their stay in the intensive care unit, such as their vital signs, and relevant information associated with their care, such as mechanical ventilation settings, laboratory tests, mental status, etc. It contains certain values repeated with the LABEVENTS table, which were included by the medical staff with the objective of unifying the information in a single table. In case of discrepancies between the values, those in the LABEVENTS table are taken as correct. It contains around 330,000 records.
- CPTEVENTS: This table contains CPT (Current Procedural Terminology), codes that identify the procedures procedures carried out in each patient. They are mainly used for billing.
- D CPT: Contains general definitions, with little detail, of CPT codes used in the CPTEVENTS table. This is an auxiliary table that does not have a unique relationship with the CPT entries, each D CPT entry corresponds to a range of codes. In this way, multiple CPT codes can share the same description, as they are similar procedures.
- D ICD: This table contains the list of diagnostic codes and their description according to the "International Coding Definitions Version 9" (ICD-9) standard. They are assigned at the end of the patient's stay and are used in billing. It contains 14567 records, each corresponding to a different diagnostic code. • DICDPROCEDURES: Similar to

the DICDDIAGNOSES table, it contains the descriptions of the procedure according to the ICD-9 standard.
- D ITEMS: Contains the description of all the elements stored as "ITEMS". Each

  It comes from the database of Philips CareVue and Metavision monitoring devices. It should be taken into account that it is possible to have duplicate elements, as they are repeated in both databases, as well as due to manual text entry and differences in spelling or punctuation. The ITEMIDs coming from the Metavision device are greater than 220000.
- D LABITEMS: This table comes from the hospital database and contains definitions for all ITEMIDs associated with laboratory measurements. It is indicated that the information contained in this table is consistent, with no duplicates present. It is externally linked to the LOINC database, which presents a universal standard for coding medical records. • DATETIMEEVENTS: Contains the record of dates

and times of events related to a patient in the ICU. To protect the identity of the patients, the dates have been shifted in time, consistent with the rest of the data, thus maintaining the chronology of the patients.
- ICD DIAGNOSES: Contains patient diagnoses coded using the ICD-9 standard. HE assigned for billing purposes at the end of each patient's hospital stay.
- DRGCODES: Contains diagnostic related groups (DRG) codes, for the patients.
- ICUSTAYS: Defines each stay in the intensive care unit. It is a table derived from grouping from the TRANSFERS table by ICUSTAY ID.
- INPUTEVENTS CV: Comes from the Philips CareVue monitoring system database and contains information about fluids administered to the patient, such as feeding tubes or intravenous solutions. • INPUTEVENTS MV: Table analogous to

INPUTEVENTS CV, containing the drugs administered to the patient registered by MetaVision.

- LABEVENTS: Contains all laboratory measurements for a given patient, including those taken in external clinics. The latter do not have a hospital admission identifier, as they have not been taken in the hospital.
- MICROBIOLOGYEVENTS: Contains microbiology records, including tests performed and sensitivities to different strains of bacteria and viruses, from patients in the ICU.
- NOTEEVENTS: Contains text notes about patients taken by medical staff. Highlights information about the patients' clinical history, as well as textual interpretations of different tests, nursing reports and notes or instructions to follow after discharge and prescribed medications.
- OUTPUTEVENTS: Contains measurements on fluids excreted by the patient during their hospital stay, such as urine, blood, sputum, etc.
- TRANSFERS: Contains the location of the patients throughout their hospital stay. From this table derives the ICUSTAYS table.
- PATIENTS: Contains information about patients, such as their sex, date of birth, or date of death, where applicable. In those patients older than 89 years, their date of birth has been modified, recording it as 300 years prior to the date of first admission. This modification is made to comply with US data protection regulations (HIPAA). The median age for these patients is 91.4 years.
- PRESCRIPTION: Contains prescriptions for drugs prescribed to patients and information related to their supply: duration, dose, ratio, etc.
- PROCEDUREEVENTS MV: Contains information about medical procedures performed on patients during your hospital stay.
- SERVICES: This table describes the services under which each patient was admitted during their stay, which may differ from the type of intensive care unit in which you are housed due to a variety of reasons, such as For example, lack of beds. The services are stored using their abbreviations according to the following table:

Service	Meaning	Description
CMED	Cardiac Medical	Non-surgical admissions for cardiac reasons
CSURG	Cardiac Surgery	Surgical admissions for cardiac reasons
DENT	Dental	Dental admissions
ENT	Ear, nose, throat	Otorhinolaryngology Admissions
G.U.	Genitourinary	Genitourinary admissions
GYN	Gynecological	Gynecological admissions
MED	Medical	General Admissions
N.B.	Newborn	Neonates
NMED	Neurological Medical	Non-surgical neurological admissions
NSURG	Neurological Surgical	Neurological Surgical Admissions
OBS	Obstetrics	Obstetrics Admissions
ORTHO	Orthopedic	Orthopedic surgical admissions
OMED	Orthopedic medicine	Orthopedic non-surgical admissions
PSURG	Plastic	Plastic/Reconstructive Surgery Admissions
PSYCH	Psychiatric	Psychiatry Admissions
SURG	Surgical	General Surgery Admissions
TRAUM	Trauma	Trauma admissions
TSURG	Thoracic Surgical	Thoracic surgery admissions
VSURG	Vascular Surgical	Non-cardiac vascular surgery admissions

2.3 Variable to predict: out-of-hospital mortality

We obtain the groups of this variable from the union of the ADMISSIONS and PATIENTS tables. Specific, The time period in months between the patient's discharge date is extracted, from the ADMISSIONS table, and the date of death of the patient, contained in the PATIENTS table. In cases where the patient does not die in the hospital, this date comes from the US social security database. In the database, this variable is distributed in the following way:

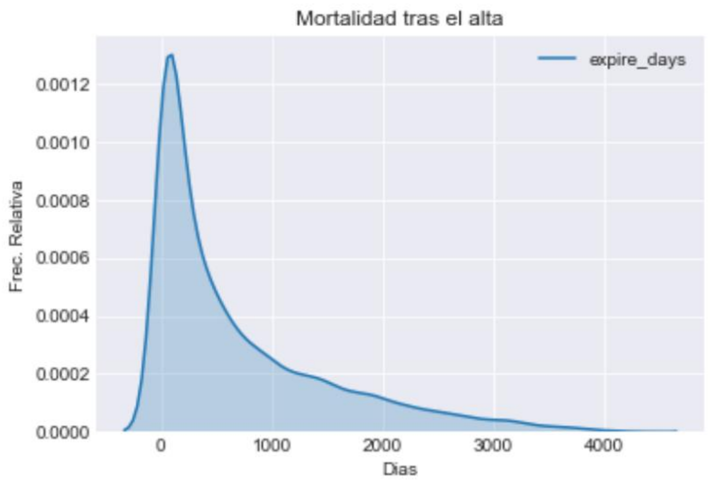


Figure 2.2: Probability distribution of mortality after discharge

Statistical descriptor	Worth
Count	16548 records
Arithmetic mean (µ)	708 days
Standard deviation (ÿ) 820 days	
Minimum value 0.5 days	
25% percentile 88 days	
50% percentile 374 days	
75% percentile 1067 days	
Maximum value	4327 days

Currently, the studies carried out in this field have not been able to obtain clinical results. significant through regression models, that is, it has not been possible so far to obtain the numerical value of the survival time of patients after discharge. This is due to the great complexity of the data and its underlying relationships. Work in this area so far has focused on binary classification tasks. or multiclass. In this way, after exploring the information and its statistics, it was decided to classify the patients into the following three survival groups.

Mortality Quantity Percent	
12+ months 8391	37%
1-12 months 6095 < 1	27%
month 8100	36%

The selection has been made expressly to avoid unbalanced classes that make subsequent prediction difficult. Likewise, it is a useful group in clinical practice.

For example, if the model is put into production and predicts that a patient has a high probability of dying in less than a month, the medical staff should consider the patient's situation and discharge. To do this, we use the following query, which directly applies the classification into groups using a CASE statement in SQL.

```
SELECT hadm_id,
CASE
WHEN
     EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) > 12
     THEN '12+ months'
     WHEN
     EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) < 12 AND
       EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) >= 1

     THEN '1-12 months'

    WHEN
     EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) < 1 AND EXTRACT(epoch
     FROM (dod-dischtime))/(3600*24*30) > -0.5 THEN '0-1 months'


    END
AS mortality
FROM admissions to
INNER JOIN patients p ON
a.subject_id = p.subject_id
```

2.4 Predictor variables

Next, a series of determining variables are compiled when predicting survival time after discharge from hospital admission. The collected variables are extracted from database queries and in some cases from their preprocessing. The distribution into classes meets only organizational criteria.

2.4.1 Demographic information

Five demographic variables are collected: Age, sex, marital status, religion and ethnicity. These variables are extracted directly from the ADMISSIONS table, except for age, which is calculated from the time difference between the date of birth, stored in the PATIENTS table, and the hospital admission date, from the ADMISSIONS table. .

Age

The age of patients over 91 years of age is shifted in time in order to protect their identity and make their identification more difficult, in compliance with the US privacy law, HIPPA. In this way, we found elderly patients with ages over 300 years. Using a preprocessing function we replaced the age of these patients with 91 years. We will later discard these records from the data set that will be used to train the neural network, considering them unreliable and prone to inducing errors. Likewise, neonates will also be discarded, as they present a very different medical behavior from that of the adult population.

```
SELECT hadm_id,
EXTRACT(epoch FROM (admittime dob))/(3600*24*365)
AS age
FROM admissions to
INNER JOIN patients p ON
a.subject_id = p.subject_id
```

It is statistically distributed in the following way

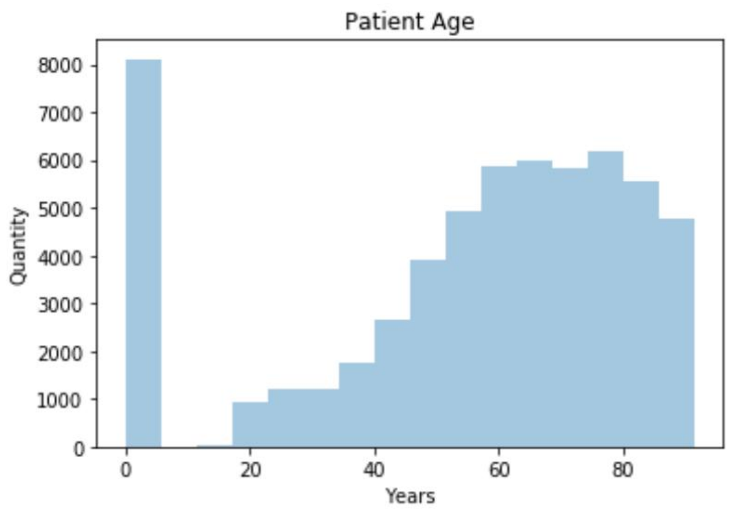


Figure 2.3: Histogram of the age distribution of the patients

Statistical descriptor	Value (years)
Count	58976
Arithmetic mean (μ)	55.2
Standard deviation (ÿ)	27.3
Minimum value	0
25% percentile	43.5
50% percentile	61.8
75% percentile	75.9
Maximum value	91.4

Sex

We extract this variable for each hospital admission directly from the database, without any type of pre-processing, using the following simple query.

```
SELECT hadm_id, gender
FROM admissions to
INNER JOIN patients p
ON a.subject_id = p.subject_id
```

It is distributed as follows

	Count	Proportion
Men	39250	55.8%
Women	26026	44.2%

Civil status

We obtain it through the following consultation, in an analogous way to the patient's sex.

```
SELECT hadm_id, marital_status
FROM admissions to
INNER JOIN patients p
ON a.subject_id = p.subject_id
```

We observed clearly unbalanced and insignificant classes that should be treated.

Civil status	Amount
DIVORCED	3213
LIFE PARTNER	fifteen
MARRIED	24239
SEPARATED	571
SINGLE	13254
UNKNOWN (DEFAULT)	345
WIDOWED	7211

To preprocess this variable, we unify those groups with similar characteristics. Specifically, they come together the DIVORCED and SEPARATED groups into one, and LIFE PARTNER is included within MARRIED. To do This grouping takes into account the lifestyle habits and social-demographic factors that may characterize each group. After making this grouping, we arrive at the following classes:

Machine Translated by Google

Civil status	Amount
DIVORCED/SEPARATED	3784
MARRIED	24254
SINGLE	13254
UNKNOWN	10473
WIDOWED	7211

Religion

We carry out a procedure analogous to that carried out in the MARITAL STATUS variable, taking into account the same considerations when unifying groups. We extract the variable from the database with the following consultation

```
SELECT hadm_id, religion
FROM admissions to
INNER JOIN patients p
ON a.subject_id = p.subject_id
```

In the same way as with marital status, we obtain unbalanced and insignificant groups. Specific, we obtain 20 groups, 14 of which have less than a thousand records, out of a total of ÿ59,000. After grouping them According to similar cultural characteristics, we arrive at the following groups:

Religion	Values
BUDDHIST/HINDU	380
CHRISTIAN	29323
JEWISH/HEBREW	5330
MUSLIM	225
NONE	23176
ORTHODOX	542

Ethnicity

Carrying out the same process as for the previous variables, we extract and unify the patient's ethnicity to each hospital admission.

```
SELECT hadm_id, ethnicity
FROM admissions to
INNER JOIN patients p
ON a.subject_id = p.subject_id
```

41 different ethnic origins are compiled, some very similar to each other. For example, a distinction is made between Hispanics depending on the country, giving rise to numerous categories with less than ten entries. The same thing happens with patients Asian and Caucasian origin. There are also records of patients of native North American origin (72 records) or native to the Caribbean (9 records). These insignificant records are grouped under the OTHER category. The result of preprocessing the variable is as follows:

Ethnicity	Amount
ASIAN	2007
BLACK	5785
HISPANIC	2136
NONE	5896
OTHER	1766
WHITE	41386

Machine Translated by Google

2.4.2 Laboratory tests

Ten common laboratory tests, routinely performed upon a patient's admission, are extracted to use them as predictor variables. For each of them, we obtain its mean value and its standard deviation, giving rise to a total of twenty variables. Each of the test results is stored in the table LABEVENTS using an identifier, ITEMID.
The relationship of ITEMIDs for laboratory tests is as follows:

Lab test	ITEMID
Urea nitrogen in blood	51066
Platelet count	51265
Hematocrit	51221
Potassium in blood	50971
sodium in blood	50983
Creatinine in blood	50912
Bicarbonate in blood	50882
White blood cell count	51301
Blood glucose	50809, 50931
Albumin in blood	50862

To obtain the average and standard deviation of each of these variables, for example for blood sodium, We perform the following query:

```
SELECT hadm_id,
avg(valuenum) AS AVG_SODIUM,
stddev(valuenum) AS STD_SODIUM,
FROM labevents
WHERE itemid = 50983
GROUP BY hadm_id
```

This function runs in a loop for all lab tests. Regarding preprocessing, we discard those values below the 1% percentile and above the 99% percentile, considering them aberrant errors or measurement errors, in addition to being insignificant. It is done through a function created for this.
After extracting the variables and treating them, we obtain the following result:

Proof	Measurements (x103)	µ	ÿ	m´ÿn.	P25%	P50%	P75%	max.	Unit
Ureic nitrogen	49.9	24.5	16.1	5.6	13.4	19.2	30.3	93	mg/24hr
Platelet count	55.8	241.1	97.9	44.9	172.3	229	297.1	595.6	K/uL
Hematocrit	55.9	33.9 23.8 6.1 3.9 4.2 3.3 4.1 5.2 8 9.1 mEq/L					36.7	58.9	%
Potassium in blood	51.8		0.4		3.9		4.4		
sodium in blood	51.8	138.7	3.1	128.7	136.9	138.9	140.8	147.8	mEq/L
Creatinine in blood	49.9	1.3	1.1	0.35	0.93	1.3 0.727	3.1	1.34	7.9 mg/dL
Bicarbonate in blood	51.8	128.7 136.9 138.9 140.8 147.8 mEq/L							
White blood cell count	55.8	241	97.9	44.9	172.3	229	297.1	595.6	K/uL
Blood glucose	49.6	131.7 32.9 78.2 1.7 2.7			110 124.2 124.3 144.3 mg/dL				
Albumin in blood	30.5	3.2	0.6	3.7 g/dL		3.2		4.7	

2.4.3 Physiological signals

In the same way that we obtain the results of laboratory tests, we extract from the database the mean and standard deviation of six physiological signals to use them as predictor variables. The measurements have been taken with two different monitoring systems, Philips CareVue and Metavision. So Likewise, the database distinguishes between measurements taken automatically and measurements taken expressly by the user. medical personnel, among other factors. This is why the same measurement has multiple identifiers.

Physiological signal	ITEMID
Heart rate	220045, 211
Respiratory rate	8113, 3603, 220210, 618
Systolic pressure	51,442,455,6701,220179,220050
Day pressure	8368,8440,8441,8555,220180,220051
Temperature	223761,678
Oxygen saturation	646, 220277

The physiological signals are recorded in the CHARTEVENTS table. We use the following query, very similar to the one used to obtain the results of laboratory tests. For example, to extract the desired values In the case of oxygen saturation we would use the following query.

```
SELECT hadm_id, avg(valuenum) AS AVG_SPO2, stddev(valuenum) AS STD_SPO2,
FROM chartevents WHERE itemid IN (646, 220277) GROUP BY hadm_id
```

We apply the same processing used previously, that is, we discard those values below the 1% percentile and those above 99%. The descriptive statistics of the average of these variables are the following:

Proof	Measurements (x103)	µ	ÿ	min.	P25%	P50%	P75%	max.	Unit
Heart rate	55.6	92.2	22.9	55.5	78.9	86.8	99.8	162	BPM
Respiratory rate	55.6	22.7	10.1	12.4	16.9	19.4	23	60	BPM
Systolic pressure	48	120.5	15.2	86.8	109.2	118.8	130.6	163.8	mmHg
Diastolic pressure	48	61	9.7	39.2	54.6	60.1	67.1	91.8	mmHg
Temperature	47.2	98.2	0.9	94.7	97.7	98.2	98.8	100.6	oF
Oxygen saturation	48	96.9	1.64	89.3	96.1	97.2	98.8	99.8	%

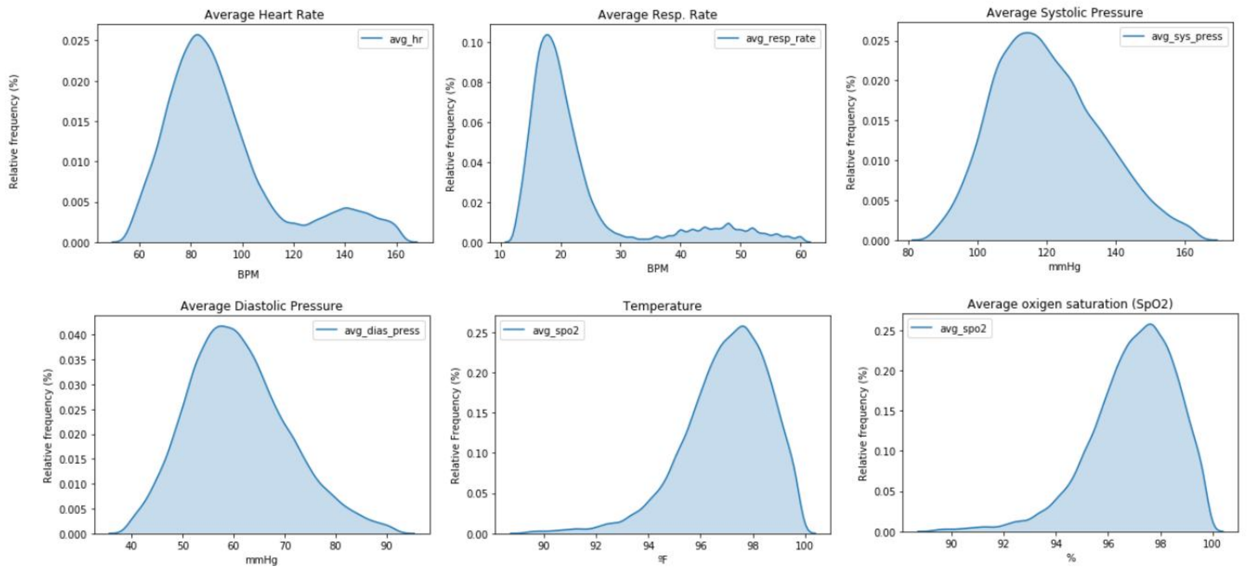


Figure 2.4: Probability distribution of the extracted physiological signals

2.4.4 Hospital information

Eleven variables related to each hospital stay are extracted.

Hospital variables	Guy
Medical service	Categorical (20 values)
ICD9 diagnostic group	Categorical
Performing surgery	Binary
Length of stay in ICU	Continuous numeric
Total length of stay	Continuous numeric
OASIS Severity Indicator	Integer numeric
SAPS severity indicator	Integer numeric
SOFA severity indicator	Integer numeric
Time in mechanical ventilation	Continuous numeric
Death in hospital	Binary
Number of procedures performed	Integer numeric

Medical service This is a categorical variable that indicates the most relevant medical service for which it is the patient was cared for during the hospital stay. Because on numerous occasions a patient remains in more than one service during its stay, a preprocessing function is necessary that extracts the service with the highest importance based on a criterion.
Specifically, a function has been designed that uses the following priority to extract a single service for each hospital stay.

- Specialized surgery services
- General surgery service
- Specialized service
- General medicine service

In this way, a patient admitted to the general medicine service and subsequently transferred to the cardiac surgery, will be recorded as a patient treated under the cardiac surgery service only, for example.
This allows us to reduce the dimensionality of the variable and obtain the most relevant information. After applying This preprocessing, the following categories and counts are obtained.

Medical service	Meaning	Amount
MED	General medicine	17260
N.B.	Neonates	7806
CSURG	Heart surgery	7697
CMED	Cardiology	5860
SURG	General Surgery	5034
NSURG	Neurological surgery	4024
TRAUM	Traumatology	2699
NMED	Neurology	2324
OMED	Obstetrics	1475
VSURG	Non-cardiac vascular surgery	1371
TSURG	Thoracic surgery	1281
ORTHO	Orthopedics	739
G.U.	Urology	334
PSURG	Plastic surgery	269
GYN	Gynecology	206

ICD-9 Diagnostic Group ICD-9 is the acronym for "International Statistical Classification of Diseases and Related Health Problems 9th Revision", published by the World Health Organization in 1977.
They are used to classify and codify pathologies, injuries, symptoms, social circumstances and external causes of diseases, in order to collect useful health information related to deaths, diseases and injuries. These codes are divided into chapters, sections, categories, subcategories and subclassifications, for example:

(1) Codes 390 – 459: Diseases of the circulatory system

(I) Cerebrovascular diseases (430-438)

(A) Occlusion of cerebral arteries (434) (i)

Cerebral embolism (434.1) (a)

Cerebral embolism with cerebral infarction (434.1.1)

The most current version is ICD-10, which was developed in 1992, although the MIMIC III v1.4 database includes the previous version, ICD-9. Currently, the widespread transition is underway worldwide from the ICD – 9 standard to ICD – 10. Due to the wide variety of codes and the class imbalance of each specific code, Only the primary code ICD – 9 is used, as shown in the following list:

• Codes 001 – 139: Infectious and parasitic diseases • Codes 140 –
239: Neoplasias
• Codes 240 - 279: Endocrine, nutritional and metabolic diseases and immune disorders
• Codes 280 – 289: Diseases of the blood and hematopoietic organs
• Codes 290 – 319: Mental disorders
• Codes 320 – 389: Diseases of the nervous system and sense organs • Codes 390 – 459: Diseases
of the circulatory system • Codes 460 – 519: Diseases of the
respiratory system
• Codes 520 – 579: Diseases of the digestive system
• Codes 580 – 629: Diseases of the genitourinary system
• Codes 630 – 679: Complications of pregnancy, childbirth and puerperium
• Codes 680 – 709: Diseases of the skin and subcutaneous tissue • Codes 710
– 739: Diseases of the osteo-myoarticular system and connective tissue • Codes 740 – 759:
Congenital anomalies
• Codes 760 – 779: Certain diseases originating in the perinatal period
• Codes 780 – 799: Symptoms, signs and ill-defined states
• Codes 800 – 999: Injuries and poisoning
• Codes E and V: External causes of injuries and supplementary classification.

Using the following query we obtain the highest priority ICD-9 code for each admission, indicated by seq num = 1 in the database.
_

```
SELECT hadm_id, diagnoses_icd.icd9_code FROM
diagnoses_icd INNER
JOIN d_icd_diagnoses ON
diagnoses_icd.icd9_code = d_icd_diagnoses.icd9_code WHERE
seq_num = 1
```

A filter function is necessary that converts the specific ICD-9 code to its highest classification based on its code number, which will be done in the preprocessed.

Performing surgery To detect whether surgical interventions have been performed on a patient during their hospital stay we will use the surgery indicators (Surgery Flags), provided by the HCUP, Health-care Cost and Utilization Project, an initiative financed by the US government through the 'Agency for Healthcare Research and Quality' (AHRQ) dedicated to the management and analysis of medical data. This entity provides tools to identify surgical interventions and events using ICD-9 codes. of procedures or CPT codes (Current Procedural Terminology), both present in the MIMIC-III database v.1.4. It allows the classification of procedures into three groups:

- NARROW: Invasive therapeutic surgical procedures requiring incision, excision, manipulation or suturing of tissue that penetrates or passes through the skin, typically performed in the operating room and with local anesthesia or general or sedation.
- BROAD: Surgical procedures that cannot be classified as those included in the NAR-ROW indicator, but are performed under surgical conditions. This group includes diagnostic surgical procedures, such as endoscopic or percutaneous procedures, or those performed through natural orifices. These are less invasive interventions.
- NEITHER: Procedures not registered as NARROW or BROAD, that is, non-surgical procedures.

This classification is distributed in the form of a CSV file and using Python a function is designed to return the classification of the procedure. Because only 4% of records are classified as BROAD, it is decided include these within NARROW in order to avoid disproportionate classes, resulting in a binary variable with the following distribution.

Surgery Indicator	Count	Percent
Narrow	29867	56%
No Surgery	23043	44%

Length of stay in ICU From the database it is possible to directly extract the length of stay in ICU in days for each patient in the same hospital admission. This information is in the ICUSTAYS table and we obtain it through the following query.

```
SELECT hadm_id,
sum(los) AS total_icu_time
FROM icustays
GROUP BY hadm_id
```

It is necessary to use the aggregate sum function in the consultation because on certain occasions a patient enters in the ICU, is transferred to another section and subsequently returns to the ICU, with which different durations are recorded for the same stay. In this way, we obtain a continuous variable, to which we do not apply preprocessing.

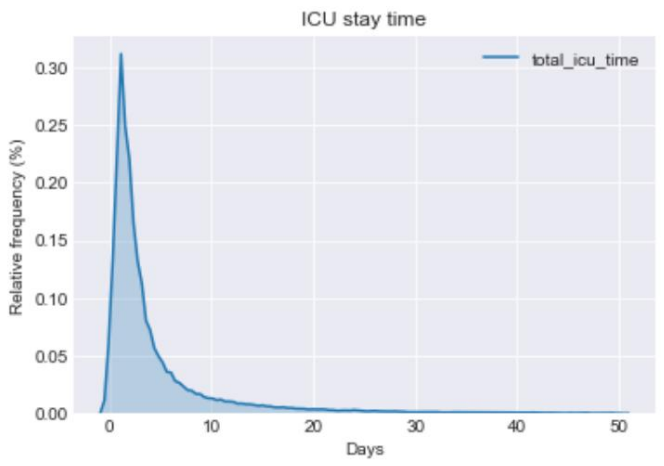


Figure 2.5: Probability distribution of ICU stay

Length of hospital stay We obtain the length of hospital stay, including the duration in the ICU, as the difference between the time of admission and discharge. To do this we use the EXTRACT and epoch functions, typical of PostgreSQL.

```
SELECT hadm_id,
EXTRACT(epoch FROM(dischtime - admittime))/(3600*24) AS total_days
FROM admissions
```

This variable is also measured in days and does not require preprocessing. It is distributed as follows.

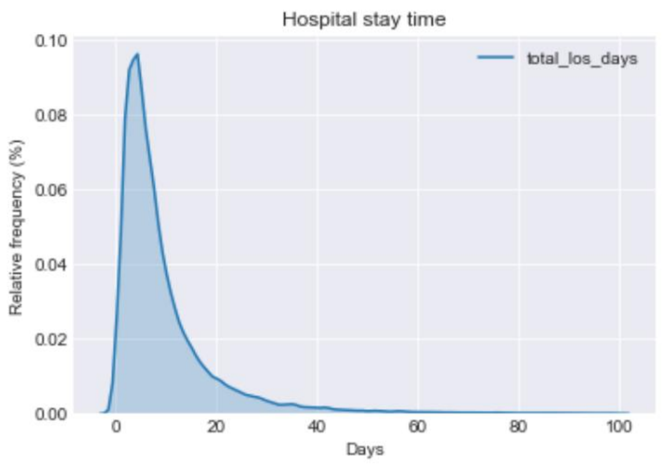


Figure 2.6: Probability distribution of the complete hospital stay

Admission count For each admission, indicates the number of hospital stays that the same patient has made, also counting the same. This allows us to identify those admissions corresponding to patients readmitted on several occasions, which may be an indicator of subjects with chronic diseases, who regularly require medical attention. We obtain this variable through preprocessing functions on the ADMISSIONS table.

Procedure count Indicates the number of procedures, both surgical and non-surgical, performed on a patient in the same hospital stay. It is a discrete numerical variable that we obtain through the following query.

```
SELECT hadm_id, count(*) AS procedure_count
FROM procedures_icd
GROUP BY hadm_id
```

This variable does not require preprocessing.

Time on mechanical ventilation Using again a materialized view available in the MIMIC-III code repository, we obtain the time that each patient spends on mechanical ventilation during their hospital stay. We use the following query on the VENTDURATIONS view.

```
SELECT hadm_id, SUM(duration_hours) AS total_mech_vent_time
FROM ventdurations v
INNER JOIN icustays i ON
v.icustay_id = i.icustay_id GROUP BY
hadm_id
```

Sometimes, a patient spends time on the mechanical ventilator, is disconnected, and is connected again later. This is why the materialized view sometimes records several entries for the same ICU stay, making it convenient to calculate the sum of durations for a stay.

Severity indicators

Different severity indicators have been developed with the objective of predicting in-hospital mortality based on patient information, particularly the measures taken during the first 24 hours of admission.
However, they present certain limitations, for example by depending on subjective measurements taken by medical personnel or by using linear relationships that do not adapt to reality.
We will use the SOFA, SAPS and OASIS indicators as predictor variables, which we will obtain through the following query:

```
SELECT o.hadm_id,
AVG(o.oasis) AS oasis_avg,
AVG(so.sofa) AS sofa_avg,
AVG(sa.saps) as saps_avg FROM
oasis o
INNER JOIN sofa so
ON o.hadm_id = so.hadm_id
INNER JOIN saps sa ON
sa.hadm_id = so.hadm_id
GROUP BY o.hadm_id
```

To obtain these indicators, we use scripts from the official MIMIC-III code repository. [https://github.com/MIT-LCP/mimic-code/tree/master/concepts/severityscores]. In this way, we create materialized views that contain the precalculated severity indicators for each hospital admission.

SOFA “Sequential Organ Failure Assessment score”. Created in 1994 by the European Society of Intensive Medicine (ESICM), this indicator was developed to evaluate the severity of the patient's disease, based on the degree of organ failure of six organs. Specifically, the following measures are taken.

- Respiratory system:
- – PaO2
  - – Presence of mechanical ventilation
- Nervous system:
- – Glasgow Coma Scale.
- Cardiovascular system:
- - Mean arterial pressure
  - – Dopamine level
  - – Epinephrine Level
  - – Norepinephrine level • Liver:
- – Bilirubin level
- Coagulation: • –

Platelet level
- Kidney:
- – Creatinine level
  - – Urine volume

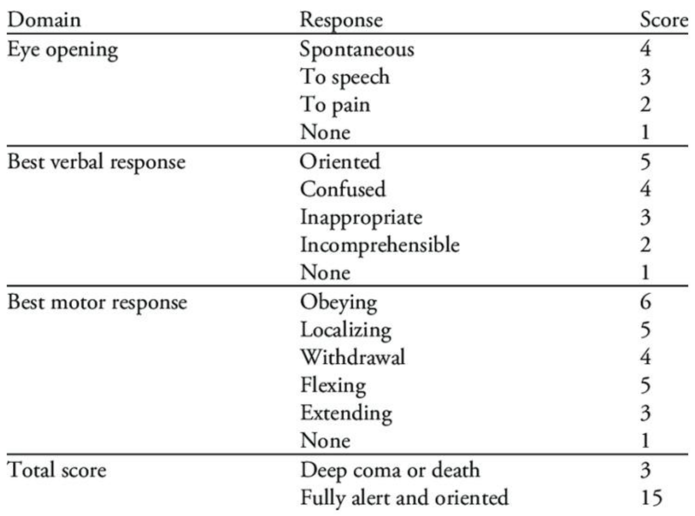
The results of these tests provide scores between 0 and 4, which are subsequently added to obtain the total SOFA score. It allows obtaining an approximate idea of the patient's mortality, in a simple and direct way to calculate from only eleven basic variables.

SAPS “Simplified Acute Physiology Score”. Created in 1993 by Le Gall and Lemenshow Saulnier, it is used to measure the severity of the disease of patients admitted to the intensive care unit over 15 years of age. It is completed 24 hours after admission and provides a score between 0 and 163, in addition to the predicted mortality in percentage. It is calculated from 12 basic physiological measurements, the patient's age and the type of admission.
The SAPS result is best used to contrast the severity of groups of patients with different pathologies, rather than at the individual level, because its results may be imprecise at the patient level.

OASIS “Oxford Acute Severity of Illness Score”, is a severity indicator designed in 2013 by Johnson AE1, Kramer AA, Clifford GD, from the University of Oxford. It is characterized by using machine learning techniques, specifically particle swarm optimization, and by not requiring a great deal of information collection work, since it requires only ten characteristics. , excluding laboratory measurements, or information on diagnoses and comorbidities.

Glasgow Coma Scale This is a neurological scale designed to easily and objectively measure a person's state of consciousness. A patient is scored according to criteria in various aspects, and the sum of scores gives a score between 3, indicating profound unconsciousness, and 14, indicating a normal state of alertness.

It is also used as a variable to calculate the OASIS, SAPS and SOFA severity indicators. It is obtained following the following criteria:



Domain	Response	Score
Eye opening	Spontaneous	4
	To speech	3
	To pain	2
	None	1
Best verbal response	Oriented	5
	Confused	4
	Inappropriate	3
	Incomprehensible	2
	None	1
Best motor response	Obeying	6
	Localizing	5
	Withdrawal	4
	Flexing	5
	Extending	3
	None	1
Total score	Deep coma or death	3
	Fully alert and oriented	15

Figure 2.7: Parameters to calculate the Glasgow Coma value

Time on mechanical ventilation Using again a materialized view available in the MIMIC-III code repository, we obtain the time that each patient spends on mechanical ventilation during their hospital stay. We use the following query on the VENTDURATIONS view.

```
SELECT hadm_id, SUM(duration_hours) AS total_mech_vent_time
FROM ventdurations v
INNER JOIN icustays i ON
v.icustay_id = i.icustay_id GROUP BY
hadm_id
```

Sometimes, a patient spends time on the mechanical ventilator, is disconnected, and is connected again later. This is why the materialized view sometimes records several entries for the same ICU stay, making it convenient to calculate the sum of durations for a stay.

3. Predictive model

Neural networks use forms of processing analogous to those of the human brain as a basis for developing algorithms capable of finding complex non-linear relationships between input and output variables. This allows the creation of predictive models capable of learning from the information provided and generalizing the results to make new predictions. Unlike other ways of creating predictive models, artificial neural networks do not impose any type of restriction on the input variables, such as their distribution. They have also demonstrated superior performance in a wide variety of tasks compared to classic machine learning methods.

Another great advantage is its ease of scaling its predictive capacity depending on the amount of information it receives, allowing greater precision by simply adding more input data. Likewise, they eliminate the need to deal extensively with the input variables of the model, and it is not necessary to carry out a large exploratory analysis to determine which variables are most relevant and which should be excluded, nor to create new variables. from combinations of those already existing.

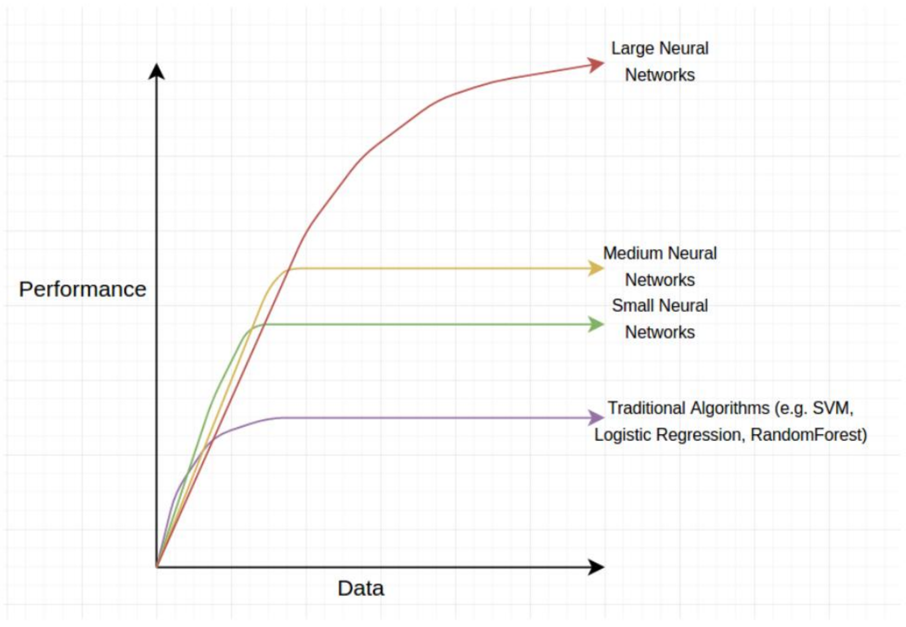


Figure 3.1: Comparison of performance of different methods to create predictive models

However, neural networks are computationally expensive and act as a black box model, making it difficult to extract and understand the relationships that the model has created. Given that the information present in the MIMIC-III v1.4 database presents complex relationships and we have an adequate volume of data, it was decided to use an artificial neural network to create a predictive model for out-of-hospital mortality.

3.1 Technologies used

To process and preprocess the information, in addition to creating the predictive model, the Python programming language, version 3.6, is intended to be used. It has been decided to use this language due to having previous experience in its use, in addition to there being a large community dedicated to its use in analysis and data mining, along with powerful libraries and extensive support. The neural network design is built with the Keras library. It is an open source library written capable of running on Tensorflow and Theano, main libraries in terms of machine learning and neural networks; offering a high-level, modular and extensible API. It includes numerous implementations of objects commonly used in building artificial neural networks, such as layers, activation functions, and optimizers. Keras was initially developed by Fran¸cois Chollet, a Google engineer, as part of research for the ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System) project in 2016. Information from MIMIC III v1.4 It is stored in a PostgreSQL database, recommended by the MIMIC-III developers. It is a free, object-oriented relational database, initially created in 1995. In the official repository of the MIMIC project there are the necessary scripts to load the CSV files, which contain the database data, on PostgreSQL, in addition to the steps for creating indexes that speed up queries. The preprocessing and numerical treatment of the data is carried out with different libraries, mainly with Pandas and Numpy, basic packages for data analysis in Python that provide data structures designed to work with relational data. in a fast, flexible and easy-to-understand way, and Scikit-Learn, a library with various machine learning tools from which we use its preprocessing functions.

Figure 3.2: Python logo

3.2 Data preparation

To obtain optimal performance of the neural network, the input data must be processed. Mainly, missing data must be treated, numerical variables normalized so that they are in similar ranges, and categorical variables deconstructed into binary variables for each of their classes.

3.2.1 Imputation of missing values (MICE)

It is common to find missing values in the MIMIC III database. For example, it is possible that a given patient has not had a laboratory test performed, or a certain physiological measure has not been taken for some reason. It is necessary that the input data of the neural network be complete in order to build the predictive model.

There are various ways to approach this problem. One possibility is to eliminate all records with a missing value, although in this case it is not viable due to the large number of variables, which means that more than 95% of records have at least one missing value, thus reducing the data set to less than 200 complete records. This amount would not allow the neural network to be trained correctly. On the other hand, missing values can also be replaced by their average or mode, although this reduces the variability of the data set and it is possible to make important errors.

Finally, it is decided to impute the missing values using the MICE (Multivariate Imputation by Chained Equations) technique. This is an iterative process that builds an imputation model for each variable through a series of regression models, based on the observed data and their existing relationships. To do this, we use the 'fancyimpute' library: [https://github.com/iskandr/fancyimpute].
After performing the imputation, the mean square error of imputation is calculated for each variable, by imputing 200 previously known values. This allows us to validate that the imputation has produced an adequate result.

Variable	% Error	Number of imputations	% Imputed values
std blood urea nitrogen std	64	9432	fifteen
platelet count std white	57	8503	14
blood cells _ _	51	8629	14
avg blood urea nitrogen avg	fifty	9036	fifteen
creatinine total icu	49	9039	fifteen
time avg platelet	46	1312	2.2
count std sodium	41	3185	5.2
_	41	8426	14
std blood glucose std	39	9541	16
temp std	37	13060	twenty-one
hematocrit std	36	8085	13
bicarbonate std hr std	34	8467	14
spo2	32	5364	8.8
avg white	32	11170	18
blood cells std potassium std	31	3158	5.2
resp rate std sys	30	8321	14
press std days	23	5482	9
press oasis avg	22	11104	18
sofa avg _	21	11134	18
_	19	1302	2.1
_	17.2	1302	2.1
saps avg	17	1302	2.1
avg albumin std	14	29669	49
creatinine	13.9	9430	fifteen
avg resp rate avg	12	3540	5.8
blood glucose avg hr	11	9376	fifteen
avg days	10	3551	5.8
press avg _	10	11075	18
bicarbonate	9.6	7190	12
avg sys press	8.9	11075	18
avg hematocrit avg	8.6	3046	5
potassium avg	5.8	7189	12
sodium avg	2	7222	12
spo2 avg	1.2	11114	18
temp	0.57	11876	19

Table 3.1: Error of imputed values

3.2.2 Normalization of numerical variables

Because the range of data values varies widely, certain functions may not work correctly.
without the normalization of these. For example, most classifiers calculate the distance between two points using
the Euclidean distance. If one of the two features has a noticeably wider range than the others, the distance
will be marked by this characteristic, so its weight in the algorithm will be much greater than it should be. By
Therefore, the range of all characteristics must be normalized in order for each variable to contribute appropriately.
balanced way. In the same way, the normalization of the variables allows the gradient descent, step
essential in a neural network, it converges at a higher speed.
Because of this, all numerical variables are normalized so that they resemble a normal distribution, where
The mean is zero and the standard deviation is one. We carry out this process using the 'scale' method of the module
'preprocessing' from the 'Scikit Learn' library

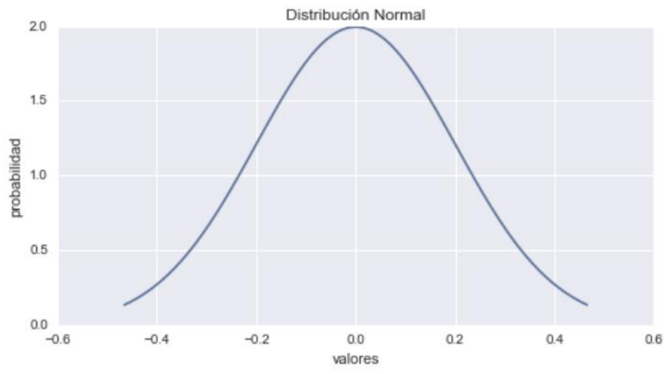


Figure 3.3: Normal probability distribution

3.2.3 Coding of categorical variables

Categorical variables must be encoded to be accepted by the neural network. Specifically, they must be divided in columns with binary values for each of the classes that the variable takes.

Patient ID	Mortality
456	< 1 month
321	1 - 12 months
678	12+ months
543	1 -12 months
987	< 1 month

For example, the table above must be converted to the following format, where each column corresponds to a range of mortality.

Patient ID	< 1 month	1 - 12 months	12+ months
456	1	0	0
321	0	1	0
678	0	0	1
543	0	1	0
987	1	0	0

We carry out this process both for the variable to be predicted, out-of-hospital mortality, and for those variables categorical predictors, such as patient ethnicity, religion, or ICD9 diagnostic code group, among other. It is carried out using the get dummies method of the Pandas numerical calculation library.

3.2.4 Division into training and evaluation set

It is convenient to divide the data set into two sets, a first larger set on which the neural network, and a smaller test set on which the performance of the network is evaluated. It is decided allocate 7.5% of the data set to the evaluation of the model, distributing the set as follows mortality data:

	Mortality
Training set	20778 records
Evaluation set	1685 records
Total	22463 records

It is done using the 'train test split' tool from the 'Scikit Learn' library.

3.3 Overfitting and overgeneralization

On certain occasions, a predictive model may fit too much to the training data set, being unable to generalize the prediction to data on which it has not been trained. On the other hand, the opposite situation can also occur, that is, one has not been able to extract significant relationships during the training process, and therefore cannot make correct predictions.

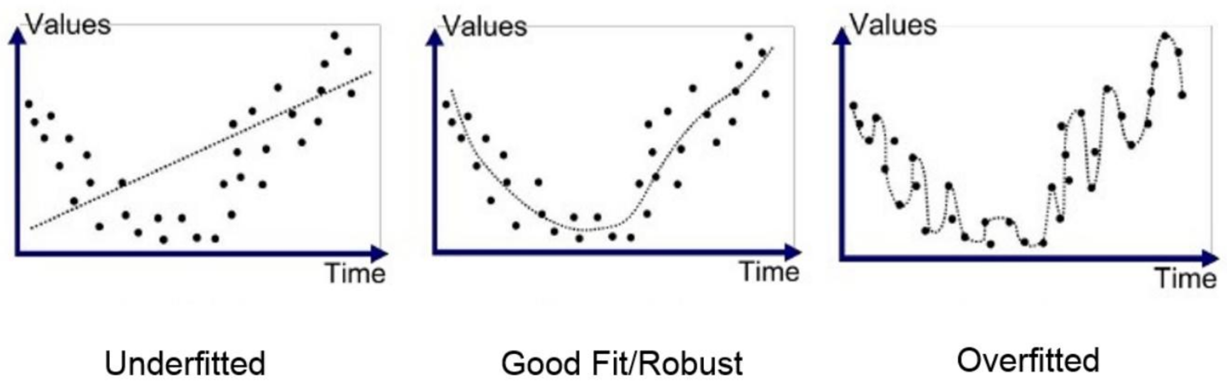


Figure 3.4: Examples of overgeneralization, adequate fit and overfitting

In this way, it is necessary to find a model that does not 'memorize' the data used during its training, but is capable of extracting significant relationships. The model with the greatest precision on the training data is not always the best, since it may be an overfitted model.

An overgeneralized model presents high error in both the training data and the test data, while an overfitting model presents very low error on the training set and high error on the test set. It will be necessary to evaluate this factor after training the neural network.

3.4 Evaluation metrics

The correct use of evaluation metrics in a classification model is essential to understand its performance.

Classification accuracy

This is the relationship between correct predictions and the total number of predictions. It is useful when the classes to be predicted are balanced, as in the case at hand. Otherwise, it usually leads to errors of interpretation. For example, in a classifying task of images from 0 to 9, if you want to build a classifier that detects the number 6, it is enough for the algorithm to classify each record as other than 6 to obtain an accuracy of 90 %, since only 10% of the images are 6. This is a major problem in machine learning tasks and that is why various metrics must be used to study the same model.

$$\text{Precision}' = \frac{\text{Correct predictions}}{\text{Total predictions}}$$

F1Score

Precision, which indicates the percentage of correct predictions by the model, and sensitivity, which indicates what percentage of records were correctly predicted, are combined in the same metric using the 'F1-Score'.
It is calculated from the harmonic mean of precision and sensitivity, and therefore will only return a high value if both sensitivity and precision are high.

$$F1 = 2\ \text{ÿ}\ \frac{\text{Precision} \cdot \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

Mean square error

In a similar way, the previous metric measures the average of the squared errors between the obtained and real values. It is calculated using the following expression and it is convenient to minimize its value. Its acronym in English is MSE (Mean Squared Error).

$$\text{MSE} = \frac{1}{n} \, {}_{t=1}^{n} \, e_t^2 \tag{3.1}$$

AUROC (Area Under Receiver Operating Characteristic)

An ROC (Receiver Operating Characteristic) curve shows the rate of true positives, the sensitivity, as a function of the rate of false positives, the specificity. Each point on the ROC curve represents the relationship between sensitivity and specificity corresponding to a given threshold value.

$$\text{Sensitivity} = \frac{\text{True positives + False negatives}}{\text{Number of samples}}$$

$$\text{Specificity} = \text{False} \; \frac{\text{False positives}}{\text{positives + True negatives}}$$

The area under this curve is a measure of the degree of fit of a predictor in a classification task. It measures the discrimination of the model, that is, the ability to correctly classify values. A model is considered to have a perfect prediction capacity when this value is 1, and random when it is 0.5.

It is the most significant evaluation measure and is commonly used to compare the performance of predictive models.

3.5 Parameters and hyperparameters

Hyperparameters are the variables that determine the structure of the neural network, such as the number of hidden layers, and the variables that determine how the network is trained, for example the learning rate.
Hyperparameters are set before training the model and determine its performance. The main hyper-parameters to select are the following:

- Number of hidden layers and units: These are the layers between the input layer and the output layer. A very high number of units in the same layer together with regularization techniques allows increasing the precision of the model. However, a low number of units can lead to overgeneralization of the model.

- Dropout: It is a technique to reduce overfitting in neural networks. It consists of randomly deactivating a percentage of neurons in each layer, in order to alternately decrease their weight in the model. In this way, the network increases its ability to generalize the results.

- Activation function: They are used to introduce non-linearities into the model, which allows the machine learning model to design non-linear prediction boundaries. The activation rectification (ReLu) function is generally used. The sigmoid function is used in the output layer in binary predictive models, while the 'Softmax' layer is used in multiclass predictive models.

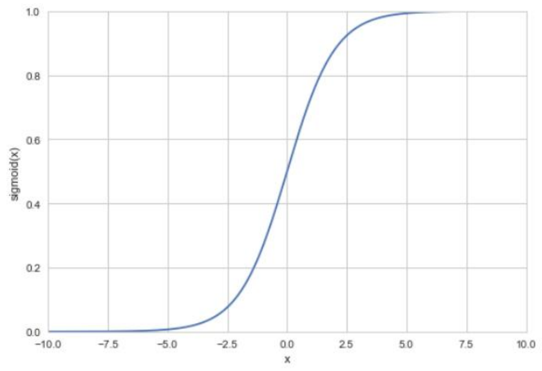


Figure 3.5: Sigmoid activation function

- Learning rate: Determines the speed of updating parameters of a network. A low learning ratio slows down the learning process, but it converges adequately. However, a high learning rate speeds up the training of the network, but may not reach the optimal parameters. This is why
They usually use adaptive learning rates, that is, high at the beginning of network training and slow a convergence is approaching.
- Moment: Allows you to deduce the direction of the next step towards the convergence of the network from the steps previous, thus avoiding oscillations on the way to the optimal parameters.
- Epochs: It is the number of times the training data set is provided to the network during the training.
- Batch Size: It is the number of samples of the training set after which the parameters of the training are updated. grid.
- Optimization algorithm: This is the algorithm used to update the parameters of the model, mainly the weights and variances of each neuron.

3.5.1 Parameter and hyperparameter adjustment

The choice of the optimal parameters of a neural network is an essential process in obtaining a good model predictive. There are various approaches to finding these values.

- Grid Search: It consists of trying all possible combinations of parameters until finding the one that minimizes the evaluation metric used. It is a less than optimal method, only useful in simple models. and quick to compile, since otherwise it requires too much time and computational resources.
- Random search: It involves randomly testing combinations of hyperparameters until you find something. that returns an appropriate result. It does not guarantee finding the optimal hyperparameters.
- Informed search: The optimal hyperparameters are searched, evaluating the result after each iteration. obtained. After each iteration, the probability of choosing values that do not correspond to the solution decreases. 'optimal. In this way, a new set of parameters is chosen, it is observed if it has been increased or decreased the quality of the model, and according to this, a new set of parameters is chosen based on the previous ones. In this way, the search for parameters is optimized, since each calculation step gets closer to the optimal parameters, in addition to not being necessary to evaluate all possible parameter combinations.

3.5.2 Bayesian Optimization

Parameter adjustment is carried out using the informed search option, specifically, we will use Bayesian optimization. It is an approach based on probabilistic models to find the minimum of a function that returns a real metric. In this case, the function is multidimensional, since it receives a space as input of hyperparameters. Bayesian optimization reduces the number of times we must train and evaluate the network neural, computationally expensive process. A probabilistic model of the objective function is constructed that looks for the correspondence between the input values, in this case the hyperparameters, and the output, the metric. assessment. In this way, the selection of hyperparameters is recalculated based on the new evidence found. after each iteration.

To implement this method we use the Hyperopt library. [https://github.com/hyperopt/hyperopt].
First we design a function that allows us to train the neural network based on the input parameters and returns an evaluation metric, the AUROC, defined above. In this case, as the optimization method Bayesian seeks to minimize the cost function, returning 1 - AUROC.
It is observed that the model receives as parameters the number of layers, the number of neurons per layer, the batch size, the number of epochs and the optimization function. We use the roc auc score module from the Scikit-Learn library to obtain the AUROC metric, since it is not included by default in Keras.
We wrap this function in an objective function, which receives only the parameters and returns the selected metric, to be used by Hyperopt:

```
deff (params):
    returntrainneur to the network (          _          , and train , X te st , and you st , params ) ;
```

Subsequently, we define the space of hyperparameters in which we will look for the optimal ones that maximize the AUROC of the model. Specifically, the following parameters are tested.

• Number of layers: 2 to 8. •

Number of neurons: 8, 16, 32, 64.

• Optimization functions: 'Stochastic Gradient Descent, Adam, RMSProp, Adagrad.

• Epochs: 10, 25, 35

• Batch size: 1, 25, 50

Hyperopt's default algorithm, TPE (Tree-structured Parzen Estimator), is used, which intelligently explores the hyperparameter space, reducing the search to the optimal ones after each iteration. After 50 iterations and an execution time of 3.5 hours, we obtain that the parameters that maximize the AUROC are the following:

• Number of layers: 6

• Neurons per layer: 8

• Batch Size: 50

• Epoch: 25 •

Optimization function: Adam

3.6 Architecture and chosen parameters

Using the parameters obtained previously through Bayesian optimization, we obtain the following network architecture, consisting of six fully connected layers with 16 neurons per layer. The input layer is a vector of 101 elements, as many as input variables after processing, although to represent it visually it is shown as a single node.

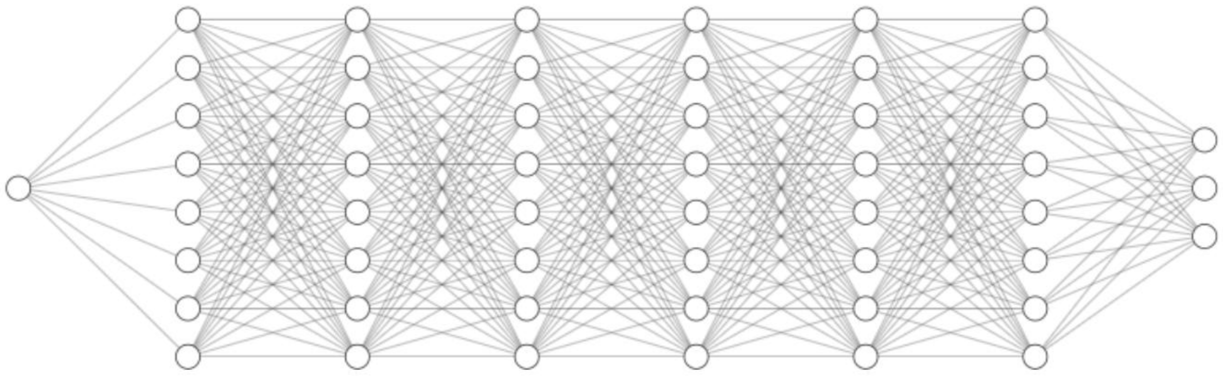


Figure 3.6: Neural network architecture

Once the network has been trained and evaluated, the need to add Dropout layers will be judged to regulate overfitting, if any. The 'ReLU' Rectified Linear Unit activation function is used. This function avoids the problem of fading gradients. The neural network calculates the weights of the parameters from the difference in the model output based on them. Certain activation functions compress the result in a certain range, for example [-1, 1] in the case of the hyperbolic tangent. In this way, when training the parameters of the first layers of the network through backward propagation, the convergence of the parameters of these layers becomes very slow, because the parameter changes produce negligible effects on the network output.
The ReLu activation function avoids this problem, since it does not restrict the output of the layer to a certain range, but rather allows values in the interval {0, ÿ}.
It also reduces the computational load based on other functions, since its calculation is simple and immediate, without requiring trigonometric or exponential functions. This allows you to train the network faster.

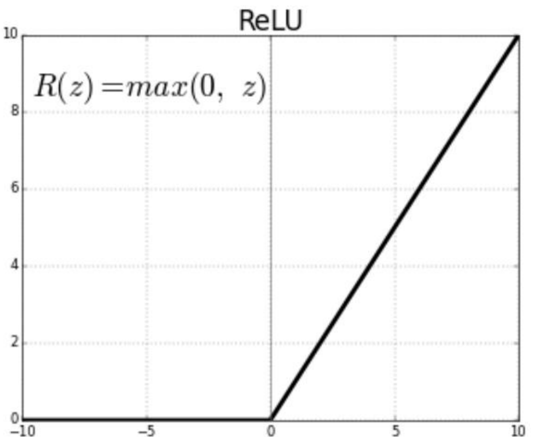


Figure 3.7: ReLu activation function

As for the optimization algorithm, we will use the 'Adam' function, based on the estimated adaptation of the moment. It was presented in 2015 by researchers at the University of Toronto. The method computes the learning rate of the network adaptively for different parameters based on estimates of the first and second moments of the gradients. It combines the advantages of other extensions of the classic method, the Stochastic Gradient Descent algorithm. Empirical results show that Adam works correctly in practice and compares favorably with other stochastic optimization methods.

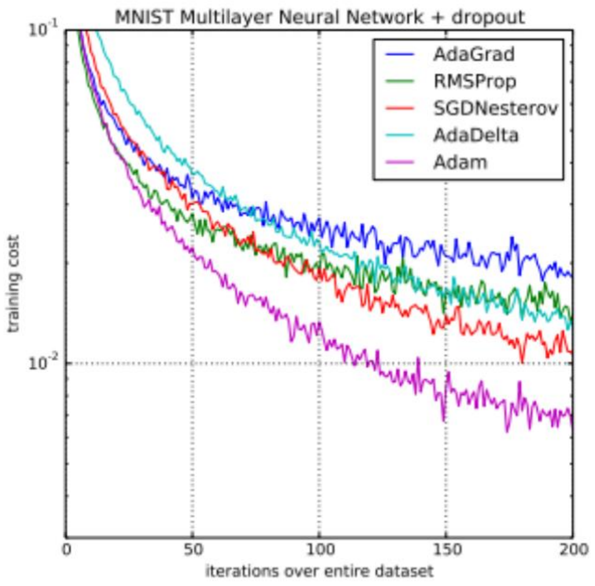


Figure 3.8: Performance comparison between optimization algorithms

On the other hand, we will make 25 passes of the data set during the training of the neural network and we will update the parameters every 50 samples, using the optimal values of epoch and batch size obtained in the previous step.

Machine Translated by Google

4. Model evaluation

Once the parameters for the network have been chosen, it is convenient to evaluate its performance based on its performance metrics. evaluation, in addition to determining whether overfitting to the training data occurs.

4.1 Results

The result of the different evaluation metrics of the model is shown below.

Metrics	Worth
Precision on training data 0.656	
Precision on test data 0.632	
Mean square error	0.154
AUROC	0.812

We observe that the network does not produce either overfitting or overgeneralization, since, as is correct, the precision over the training data is slightly higher than the precision on the evaluation data.
We separately calculate the AUROC for each of the three predicted classes and plot their curves:

AUROC Mortality Class		
0	< 1 month	0.89
1	1 - 12 months 0.72	
2	> 12 months 0.81	

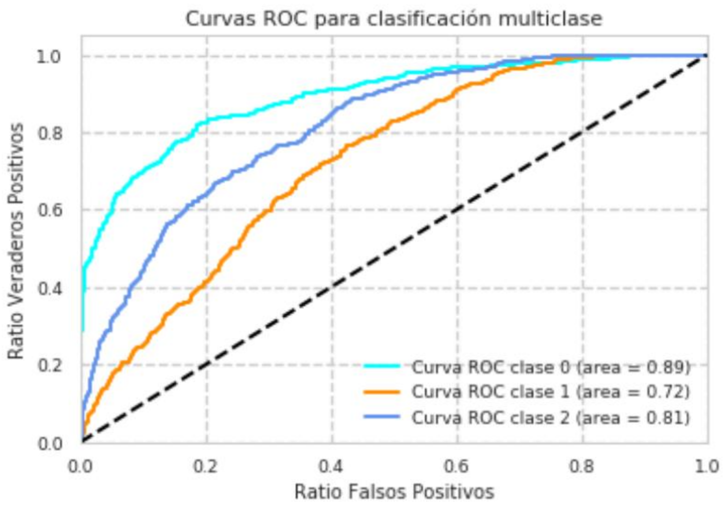


Figure 4.1: AUROC curves

The combined AUROC of the three classes obtained indicates that it is a good model, with good capacity diagnosis, especially in the class with the greatest medical value, that of patients who die within a month of his hospital discharge. We obtain a good predictive capacity for the class older than one year, although moderate for the group between 1 and 12 months.
Regarding the normalized confusion matrix of the model, we obtain the following result:

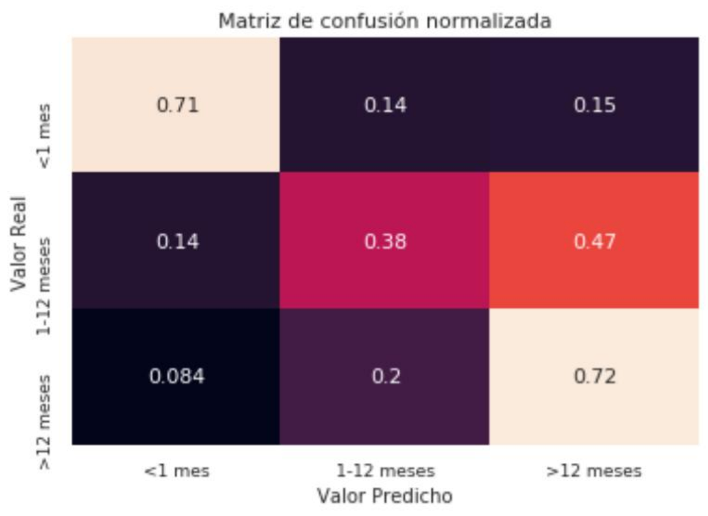


Figure 4.2: Confusion matrix

Again, it is observed that the model has difficulties in discerning patients in the interval of 1 to 12 months, although It does not do so for the other two classes. The evaluation metrics report for the three classes is as follows.

Class	Precision	Recall	F1 Score
<1	0.79	0.71	0.75
month 1 - 12 months	0.47	0.38	0.42
>12 months	0.57	0.72	0.64

4.2 Example of use

To use the model to make predictions, the following procedure must be performed. Firstly you have to provide categorical information related to the patient and their stay through an object:

```
patient_categorical_features = {'gender': 'M',
                                'marital_status':'SINGLE',
                                'religion':'CHRISTIAN',
                                'ethnicity':'WHITE',
                                'service':'CSURG',
                                'icd9_group':'diseases of the circulatory system',
                                'SURGERY_FLAG':'NARROW'
                               };
```

And in two other objects, the results of laboratory tests and physiological signals. Each variable must be entered as an arrangement of elements, from which the average and standard deviation are subsequently calculated. The units of each have also been described in the 'Predictor variables' section.

```
patient_lab_tests = {'blood_urea_nitrogen': [23, 24, 24],
                     'platelet_count':[230, 240],
                     'hematocrit':[33, 35],
                     'potassium': [3.9,3.8,4.4],
                     'sodium':[140, 139],
                     'creatinine':[1.3,1.2,1.3],
                     'bicarbonate':[25,26],
                     'white_blood_cells':[8.5,9,13],
                     'blood_glucose':[130, 135,140],
                     'albumin':[3.5, 3.4] };
```

```
patient_physio_measures = {'heart_rate':[100,108,105,99], 'resp_rate':
                                        [22,25,23], 'sys_press':[120,
                                        121, 115], 'days_press':[70,80,85] ,
                                        'temp':[98, 98.2, 97.8], 'spo2':
                                        [97,97.8,98] };
```

Subsequently, we will use the preprocess prediction data function, to which we will supply the patient's characteristics in object form, the numerical variables with previously imputed missing values, and the categorical variables. In this way, if there are missing values, for example if the patient's temperature has not been taken, we can impute them using the MICE method previously defined.

```
prediction_data = NeuralNetworkService().preprocess_prediction_data( patient_features,

    imputed_numerical_features,
    categorical_features
);
```

This function returns the predictive variables of the treated patient to be introduced into the model. Specifically, numerical variables have been normalized, missing values imputed, and numerical variables categorized.
Using these values, we make the prediction below:

```
prediction = keras_model.predict(prediction_data);
```

For this patient, the prediction result returns the following probabilities per class:

< 1 month	1 - 12 months	> 12 months
0.54086	0.27766	0.18148

Seeing this result, medical personnel should evaluate the decision to discharge the patient, due to his high risk of dying within one more time after leaving the hospital.

4.3 Performance comparison

In this section, the predictive capacity of the model developed here is compared with other studies in the field of prediction of medical destinations.

Mortality prediction in intensive care units with the Super ICU Learner Algorithm (SICULA): a population-based study. This study uses various combined machine learning techniques to predict in-hospital mortality in critically ill patients. It uses version II of the MIMIC database, earlier than the one used here. It uses the 17 variables used by the SAPS II and SOFA severity indicators, which it far surpasses in predictive capacity. Specifically, it achieves an AUROC of 0.94 for the prediction of hospital mortality. In this way, it has a higher predictive power than the neural network designed in this work, although it is the prediction of in-hospital rather than out-of-hospital mortality.

- Pirracchio R, Petersen ML, Carone M, Rigon MR, Chevret S, van der LAAN MJ. Mortality prediction in the ICU: can we do better? Results from the Super ICU Learner Algorithm (SICULA) project, a population-based study. The Lancet Respiratory medicine. 2015;3(1):42-52. doi:10.1016/S2213-2600(14)70239-5.

Mortality prediction with self normalizing neural networks in intensive care unit patients This study, carried out by researchers at the University of Waterloo, Canada, in April 2018, focuses on the prediction of in-hospital mortality and 30-day mortality , also studied in this work. It uses the MIMIC II database and evaluates the model on 17,000 patients, obtaining an AUROC of 0.84 for 30-day mortality and 0.86 for in-hospital mortality. To do this, it uses a self-normalizing neural network. These networks are characterized by maintaining the normalization of the activations during their propagation through the layers of the network, for which the activation function 'SELU' (Scaled Exponential Linear Units) is used.). The AUROC achieved through this technique is lower than that obtained in this work for the same 30-day mortality prediction task. (0.84 vs. 0.89)

- MAH Zahid and J. Lee, ”Mortality prediction with self-normalizing neural networks in intensive care unit patients,” 2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), Las Vegas, NV, 2018, pp. 226-229. doi: 10.1109/BHI.2018.8333410

A machine learning approach to predicting short-term mortality risk for patients starting chemotherapy. In this document, Gradiend Boosted Decision Trees are used on a data set consisting of around 27,000 patients who received chemotherapy between 2004 and 2014 at the Dana-Farber/Brighan and Woman's Cancer Center', in Boston, Massachusetts.

A model capable of predicting mortality 30 days after the start of chemotherapy treatment is designed using demographic information, medications received, comorbidities, ICD-9 diagnosis groups, vital signs, laboratory results and data extracted from medical notes, obtaining an AUROC of 0.94, higher than that obtained in this work for the same time interval.

- A machine learning approach to predicting short-term mortality risk for patients starting chemotherapy. Ravi Bharat Parikh, Aymen Elfiky, Maximilian J. Pany, and Ziad Obermeyer Journal of Clinical Oncology 2017 35:15 suppl, 6538-6538

5. Conclusions

A predictive model for out-of-hospital mortality with good diagnostic capacity has been successfully developed. This project has involved the exploration of a large database with many different types of values: numeric, categorical, dates, text fields, etc., and the treatment of them to be processed by the neural network.

In comparison with previous studies, the model obtains a good result and corresponds to the current state of the art in this field.
The variables necessary to make predictions with this model are easy to obtain and are collected routinely, which would allow widespread use of the algorithm in medical practice, although always considering that it makes certain inaccuracies. It could be used as a second medical opinion, to confirm or cancel the discharge of patients from the intensive care unit.

The greatest workload has fallen on the selection, extraction and preprocessing of the variables used by the model. This process has led to the discarding of certain variables, for example central venous pressure, due to the availability of few samples.
On the other hand, thanks to the Bayesian optimization of hyperparameters it has been relatively simple, although slow, to find the network configuration that ensures an optimal result.

5.1 Future work

As improvements, new variables ignored in this study could be extracted. For example, medical notes are stored in the database as a free text field for each patient, which includes notes made by medical staff about the patients' history or particular medical situations. The extraction of these characteristics would involve extensive text mining work that falls outside the scope of this project, but it could greatly improve the predictive capacity of the model designed here.

Likewise, it would be possible to develop a small interface or web page that allows predictions to be made using the designed model. This interface would allow medical personnel to enter the necessary variables and return the patient's probability of mortality.

6. Economic analysis

The largest economic cost of this project falls on the salary of the engineer who developed the neural network that will carry out the mortality prediction. In this way, most of the economic cost will go to paying for your work hours, and another small part will go to covering the indirect cost of the project, consisting mainly of the computer equipment you use.

6.1 Direct costs

These are those costs that can be directly identified with a cost object, such as materials or labor destined for the project.

In this case, the direct cost is entirely the salary of the staff, whom we consider to be a self-employed worker who bills by the hour.
We consider that the project has been carried out by a single developer, with a billing of 30e/h and a dedication of 320h, thus giving a total direct cost of 9600e.

6.2 Indirect costs

The indirect cost of the project corresponds to the computer equipment used.
There are no costs associated with the software, since Python is a free open source programming language and the computer's operating system is Elementary OS loki 0.4.1, a Linux distribution based on Ubuntu and also free .

As for hardware, a laptop of the following model is used:

- Toshiba Satellite Pro A50-D-1FZ Intel Core i7-7500U/8GB/256GB SSD/15.6”

Its cost is 695e. Considering a amortization period of three years and a use period of four months, the indirect cost of hardware is 77.22 e.
In this way, the total cost of the project computed as the sum of direct and indirect costs is around 9700 e.

7. Environmental impact analysis

The environmental impact of the development of the study is negligible, consisting only of the electricity consumed by the computer.

On the other hand, considering that the predictive model developed here is implemented in medical practice and allows the premature death of 5% of patients to be prevented, we can estimate the environmental impact of the project based on contamination. on that they generate.

According to data from the INE, in 2014 each Spaniard generates 459 kilos of waste and 5.08 metric tons of carbon dioxide annually. Likewise, it consumes an average of 132 liters of water per day, which is around 48 tons of water per year.

Every year 1200 patients are admitted to the intensive care unit of the Vall d'Hebron Hospital in Barcelona.

In this way, the annual environmental impact of the project, if the predictive model is implemented in such a hospital and extending the life of 60 of the patients, would be the following:

- 27.5 tons of waste
- 305 tons of CO2
- 2900 tons of water

8. Bibliography

- Pirracchio R, Petersen ML, Carone M, Rigon MR, Chevret S, van der LAAN MJ. Mortality prediction in the ICU: can we do better? Results from the Super ICU Learner Algorithm (SICULA) project, a population-based study. The Lancet Respiratory medicine. 2015;3(1):42-52. doi:10.1016/S2213-2600(14)70239-5.
- Aya Awad, Mohamed Bader-El-Den, James McNicholas, Jim Briggs. Early hospital mortality prediction of intensive care unit patients using an ensemble learning approach. .Int J Med Inform. 2017 Dec; 108: 185–195. Published online 2017 Oct 5. doi: 10.1016/j.ijmedinf.2017.10.002
- MIMIC Code Repository. Cambridge: Laboratory for Computational Physiology, Massachusetts Institute of Technology; 2018 [updated March 15, 2018]. Available at: http://github.com/MIT-LCP/mimic-code. Consulted on May 15, 2018.
- Johnson AE, Pollard TJ, Shen L, et al. MIMIC-III, a freely accessible critical care database. SciData. 2016;3:160035. [http://dx.doi.org/10.13026/C2XW26] (2016)
- Azur MJ, Stuart EA, Frangakis C, Leaf PJ. Multiple Imputation by Chained Equations: What is it and how does it work? International journal of methods in psychiatric research. 2011;20(1):40-49. doi:10.1002/mpr.329. • Pastur-Romay LA,

Cedr´on F, Pazos A, Porto-Pazos AB. Deep Artificial Neural Networks and Neuromorphic Chips for Big Data Analysis: Pharmaceutical and Bioinformatics Applications. Gonz´alez-D´ÿaz H, Todes-chini R, Pazos Sierra A, Arrasate Gil S, eds. International Journal of Molecular Sciences. 2016;17(8):1313. doi:10.3390/ijms17081313. • Moreno RP, Metnitz PGH, Almeida E, et al.

SAPS 3--From evaluation of the patient to evaluation of the intensive care unit. Part 2: Development of a prognostic model for hospital mortality at ICU admission. Intensive Care Med. 2005;31:1345–55. [https://www.ncbi.nlm.nih.gov/pubmed/16132892]

- Awad, Aya et al. “Early hospital mortality prediction of intensive care unit patients using an ensemble learning “approach.” International journal of medical informatics 108 (2017): 185-195.
- Knaus WA, Wagner DP, Draper EA, et al. The APACHE III prognostic system. Risk prediction of hospital mor- quality for critically ill hospitalized adults. Chest. 1991;100:1619– 36. [https://www.ncbi.nlm.nih.gov/pubmed/1959406]
- Cook NR. Use and misuse of the receiver operating characteristic curve in risk prediction. Circulation. 2007;115:928–35. [https://www.ncbi.nlm.nih.gov/pubmed/17309939]
- Knaus WA, Harrell FE, Fisher CJ, et al. The clinical evaluation of new drugs for sepsis: a prospective analysis based on survival analysis. JAMA 1993; 270: 1233-41.
- WG Baxt. Application of artificial neural networks to clinical medicine. Lancet, 346 (2007), pp. 1135-1138 • PO Lang,

D. Zekry, JP Michel, M. Drame, JL Novella, D. Jolly, *et al. Early markers of prolonged hospital stay in demented in patients: a multicentre and prospective study. J Nutr Health Aging, 14 (2010), pp. 141-147- • SW Meldon, LC Mion,

RM Palmer, BL Drew, JT Connor, LJ Lewicki, *et al. A brief risk-stratification tool to predict repeat emergency department visits and hospitalizations in older patients discharged from the emergency department. Acad Emerg Med, 10 (2003), pp. 224-232

- Lokhandwala S, McCague N, Chahin A, Escobar B, Feng M, Ghassemi MM, Stone DJ, Celi LA. One-year mortality after recovery from critical illness: A retrospective cohort study. PLoS ONE, 13(5):e0197226, May 2018.

Annexes

8.1 Code used

The most relevant code fragments used in this study are shown below. The complete code is found in the project's Github repository: [https://github.com/DanielSola/mimic-iii-project]

Hyperparameter adjustment and model training Determination of the optimal parameters using the Hyperopt library and model training.

```
from services.plotting_service import * from
services.neural_network_service import * from
services.preprocessing_service import * from
features.get_features import * from
keras.optimizers import SGD, Adam, RMSprop, Adagrad from hyperopt import
hp, Trials, fmin, tpe

#Query and preprocess data nn_data
= NeuralNetworkService().get_nn_data(); categorical_features =
features().get_categorical_features(); numerical_features =
features().get_numerical_features(); imputed_numerical_features =
impute_missing_values(numerical_features);

#Split into training and test set
X_train = nn_data['mortality_data']['X_train'];
X_test = nn_data['mortality_data']['X_test'];
Y_train = nn_data['mortality_data']['Y_train'];
Y_test = nn_data['mortality_data']['Y_test'];

#Hyperparameter tuning by Bayesian optimization def f(params):
return
    train_neural_newtork(X_train, Y_train, X_test, Y_test, params);

params_space = {'n_layers': hp.choice('n_layers', range(2,8)), 'n_neurons':
                    hp.choice('n_neurons', [8, 16, 32, 64]), 'optimizer': hp.choice('optimizer',
                    ['SGD', 'Adam', 'RMSprop', 'Adagrad']), 'epochs': hp.choice('epochs', [10, 25, 35]), 'batch_size ':
                    hp.choice('batch_size', [1, 25, 50])

               };

trials_mse = Trials()

best = fmin(fn=f, space=params_space, something=tpe.suggest, max_evals=50, trials=trials_mse);

# Training neural network with optimal parameters params =
{'n_layers':6, 'n_neurons':16,
          'optimizer': 'Adam',
          'epochs':50, 'batch_size':50 };



keras_model = NeuralNetworkService().train_neural_network(X_train, Y_train, X_test, Y_test, params);
```

Making predictions We use the pre-trained model and pre-process the patient's data in order to use the model to predict their out-of-hospital mortality.

```
#Define patient features
patient_categorical_features = {'gender': 'M',
                                                         'marital_status':'SINGLE',
                                                         'religion':'CHRISTIAN',
                                                         'ethnicity':'WHITE',
                                                         'service':'CSURG',
                                                         'icd9_group' :'diseases of the circulatory system',
                                                         'SURGERY_FLAG':'NARROW' };


patient_numerical_features = {'age':80,
                                                    'total_icu_time':10,
                                                    'total_los_days':12,
                                                    'admissions_count':3,
                                                    'procedure_count':4,
                                                    'oasis_avg':40,
                                                    'sofa_avg':7,
                                                    'saps_avg':20,
                                                    'gcs':9,
                                                    'total_mech_vent_time ':130 };


patient_lab_tests = {'blood_urea_nitrogen': [23, 24, 24], 'platelet_count':
                                   [230, 240], 'hematocrit': [33, 35],
                                   'potassium': [3.9,3.8,4.4],
                                   'sodium' :[140, 139], 'creatinine':
                                   [1.3,1.2,1.3], 'bicarbonate':
                                   [25,26], 'white_blood_cells':
                                   [8.5,9,13], 'blood_glucose':
                                   [130, 135,140] , 'albumin':[3.5, 3.4] };


patient_physio_measures = {'heart_rate':[100,108,105,99], 'resp_rate':
                                            [22,25,23], 'sys_press':[120,
                                            121, 115], 'dias_press':[70,80,85], '
                                            temp':[98, 98.2, 97.8], 'spo2':
                                            [97,97.8,98] };


patient_features = {
            'patient_categorical_features': patient_categorical_features,
            'patient_numerical_features': patient_numerical_features, 'patient_lab_tests':
            patient_lab_tests, 'patient_physio_measures':
            patient_physio_measures };


# Preprocess prediction data and predict
prediction_data = NeuralNetworkService().preprocess_prediction_data(patient_features, imputed_numerical_features, prediction =
keras_model.predict(prediction_data);
```

Definition of the neural network Code used to train the neural network using the Keras library.

```
def train_neural_network(self, X_train, Y_train, X_test, Y_test, params):

    n_layers = params['n_layers']; n_neurons
    = params['n_neurons']; batch_size =
    params['batch_size']; epochs = params['epochs'];
    optimizer = params['optimizer'];


    #Model definition
    model = Sequential(); for i in
    range(n_layers):
        model.add(Dense(n_neurons, activation='relu', input_shape=(101,))) model.add(Dense(3,
    activation='softmax')); model.compile(loss='binary_crossentropy',
    optimizer=optimizer, metrics=['accuracy', 'mse'])
              model.fit(X_train,
              Y_train,epochs=epochs,
    batch_size=batch_size, verbose=1);

    #Evaluating model
    Y_pred = model.predict(X_test); Y_train_pred
    = pd.get_dummies(model.predict(X_train).argmax(axis = 1)); Y_test_pred =
    pd.get_dummies(Y_pred.argmax(axis = 1));

    #Evaluation metrics logs
    train_accuracy = accuracy_score(Y_train, Y_train_pred); test_accuracy =
    accuracy_score(Y_test, Y_test_pred); f1_score_model =
    f1_score(Y_test_pred, Y_test, average = 'samples'); loss, accuracy, mse = model.evaluate(X_test,
    Y_test,verbose=0) roc_auroc = roc_auc_score(Y_test, Y_pred); print('TRAIN
    ACCURACY:',train_accuracy, 'TEST
    ACCURACY:',test_accuracy, 'F1 SCORE:',
            f1_score_model, 'MSE:',mse, 'AUROC:',
            roc_auroc);



    return model;
```

Normalization of numerical variables Adjustment of numerical variables to ÿ = 1 and µ = 0 using the scale method of the preprocessing module of 'Scikit-Learn'

```
def scale_numerical_features(imputed_numerical_features):
    scaled_numerical_features = pd.DataFrame(preprocessing.scale(imputed_numerical_features)); scaled_numerical_features.columns
    = imputed_numerical_features.columns; scaled_numerical_features.index =
    imputed_numerical_features.index;

    return scaled_numerical_features;
```

Extraction of the most relevant medical service Preprocessing function that is applied to a Pandas DataFrame coming from the database query of the medical services that a patient undergoes during their hospital stay. Returns the one with the greatest relevance as defined above.

```
def get_relevant_admission_service(hadm_id, services_df):

    admission_services = list(services_df.loc[services_df['hadm_id'] == hadm_id].curr_service)
    ##Special surgery
    special_surgery_service = list(filter(lambda x: 'SURG' in x and x != "SURG", admission_services))
    ##General surgery
    general_surgery_service = list(filter(lambda x: x == 'SURG', admission_services))
    ##Not general medicine
    specialized_service = list(filter(lambda x: x != 'MED' and not 'SURG' in x, admission_services))
    ##General medicine
    general_medicine_service = list(filter(lambda x: x == 'MED', admission_services))

    if special_surgery_service: return
        (hadm_id, special_surgery_service[0]); if general_surgery_service:
    return (hadm_id,
        general_surgery_service[0]);
    if specialized_service:
        return (hadm_id, specialized_service[0]); if
    general_medicine_service: return
        (hadm_id, general_medicine_service[0]);
```

Imputation of missing values Using the MICE method of the fancyimpute library to impute missing values.

```
def impute_missing_values(numerical_features):
    imputed_numerical_features = pd.DataFrame(MICE().complete(numerical_features));
    imputed_numerical_features.columns = numerical_features.columns;
    imputed_numerical_features.set_index(numerical_features.index, inplace = True);
```

Elimination of outliers Function that receives a DataFrame from Pandas and eliminates the outliers between 'low quantile' and 'high quantile' of the column specified by 'column index'.

```
def remove_outliers(data, column_index, low_quantile, high_quantile):

    low_quantile_value = float(data.iloc[:, column_index].quantile(low_quantile)); high_quantile_value =
    float(data.iloc[:, column_index].quantile(high_quantile));

    data.iloc[:,column_index] = data.iloc[:,column_index].apply( lambda x: x if (x <=
            high_quantile_value and x >= low_quantile_value) else np.NaN);

    return data;
```

Database queries Below are the queries used to extract information from the database, in PostgreSQL syntax.

```
AGE_QUERY = """SELECT hadm_id, EXTRACT(epoch FROM (admittime - dob))/(3600*24*365)
                    AS age
                    FROM admissions to
                    INNER JOIN patients p ON
                    a.subject_id = p.subject_id"""

GENDER_QUERY = """SELECT hadm_id, gender
                    FROM admissions to
                    INNER JOIN patients p ON
                    a.subject_id = p.subject_id"""


MARITAL_STATUS_QUERY = """SELECT hadm_id, marital_status
                          FROM admissions to
                          INNER JOIN patients p ON
                          a.subject_id = p.subject_id"""

RELIGION_QUERY = """SELECT hadm_id, religion
                     FROM admissions to
                     INNER JOIN patients p ON
                     a.subject_id = p.subject_id"""

ETHNICITY_QUERY = """SELECT hadm_id, ethnicity
                     FROM admissions to
                     INNER JOIN patients p ON
                     a.subject_id = p.subject_id"""



SERVICE_QUERY = """SELECT hadm_id, curr_service
                     FROM services
                     ORDER BY hadm_id ASC"""

DIAG_ICD9_CODES_QUERY = """SELECT hadm_id, diagnoses_icd.icd9_code FROM
                            diagnoses_icd WHERE
                            seq_num = 1"""

PROC_ICD9_CODES_QUERY =  """ SELECT hadm_id, procedures_icd.icd9_code
                            FROM procedures_icd
                            INNER JOIN d_icd_procedures ON
                            procedures_icd.icd9_code = d_icd_procedures.icd9_code WHERE hadm_id is
                            not null AND seq_num = 1"""


ICU_LOS_QUERY = """SELECT hadm_id, sum(los) AS total_icu_time
                    FROM icustays
                    GROUP BY hadm_id
                    ORDER BY hadm_id"""

TOTAL_LOS_QUERY = """SELECT hadm_id,
                      EXTRACT(epoch FROM(dischtime - admittime))/(3600*24) AS total_days
                      FROM admissions"""

PREVIOUS_ADMISSIONS_QUERY = """SELECT hadm_id, subject_id, admittime
                                FROM admissions
                                ORDER BY admittime ASC"""
```

```
PROCEDURE_COUNT_QUERY = """SELECT hadm_id, count(*) AS procedure_count
                           FROM procedures_icd
                           GROUP BY hadm_id"""

MECHANICAL_VENTILATION_TIME_QUERY = """SELECT hadm_id, SUM(duration_hours) AS total_mech_vent_time
                                       FROM ventdurations v
                                       INNER JOIN icustays i ON
                                       v.icustay_id = i.icustay_id GROUP BY
                                       hadm_id"""

SEVERITY_SCORES_QUERY = """SELECT o.hadm_id,
                           AVG(o.oasis) AS oasis_avg,
                           AVG(so.sofa) AS sofa_avg,
                           AVG(sa.saps) as saps_avg FROM
                           oasis o
                           INNER JOIN sofa so
                           ON o.hadm_id = so.hadm_id
                           INNER JOIN saps sa ON
                           sa.hadm_id = so.hadm_id
                           GROUP BY o.hadm_id"""

GLASGOW_COMA_SCALE_QUERY = """SELECT hadm_id, AVG(gcs) AS GCS
                              FROM pivoted_gcs gcs
                              INNER JOIN icustays i ON
                              gcs.icustay_id = i.icustay_id GROUP by
                              hadm_id"""

MORTALITY_QUERY = """SELECT hadm_id,
                     CASE
                     WHEN
                              EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) >= 12
                     THEN '12+ months'
                     WHEN
                               EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) < 12 AND
                               EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) >= 1

                     THEN '1-12 months'
                     WHEN
                               EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) < 1 AND
                               EXTRACT(epoch FROM (dod-dischtime))/(3600*24*30) > -0.5
                     THEN '0-1 months'

                     END
                     AS mortality
                     FROM admissions to
                     INNER JOIN patients p ON
                     a.subject_id = p.subject_id WHERE
                  """ p.expire_flag = 1


ADMISSION_DATA_QUERY = """SELECT hadm_id, p.subject_id, admittime, dischtime
                          FROM admissions to
                          INNER JOIN patients p ON
                          a.subject_id = p.subject_id WHERE
                          EXTRACT(epoch FROM (a.admittime - p.dob)) / (365 * 24 * 3600) BETWEEN 18 AND 80 ORDER BY subject_id ASC"""
```

```
MORTALITY_TIME_QUERY = """SELECT hadm_id, EXTRACT(epoch FROM (dod-dischtime))/(3600*24) AS expire_days
                                          FROM admissions to
                                          INNER JOIN patients p ON
                                          a.subject_id = p.subject_id WHERE
                                          EXTRACT(epoch FROM (dod-dischtime))/(3600*24) > 0.5 AND p.expire_flag =
                                          1"""

HOSPITAL_EXPIRE_FLAG_QUERY = """SELECT hadm_id, hospital_expire_flag
                                                    FROM admissions"""
```

ICD-9 Diagnostic Code Grouping Function that returns the main ICD9 diagnostic code group from the detailed diagnostic code.

```
def group_diag_icd9_code(icd9_code):
    if not 'V' in icd9_code and not 'E' in icd9_code:
        truncated_code = int((icd9_code)[0:3]); if truncated_code
        > 0 and truncated_code <= 139: return 'infectious and parasitic
            diseases'; if truncated_code > 139 and truncated_code <=
        239:
            return 'neoplasms';
        if truncated_code > 239 and truncated_code <= 279:
            return 'endocrine, nutritional and metabolic diseases, and immunity disorders';
        if truncated_code > 279 and truncated_code <= 289:
            return 'diseases of the blood and blood-forming organs';
        if truncated_code > 289 and truncated_code <= 320:
            return 'mental disorders';
        if truncated_code > 320 and truncated_code <= 389:
            return 'diseases of the nervous system and sense organs';
        if truncated_code > 389 and truncated_code <= 459:
            return 'diseases of the circulatory system'; if truncated_code
        > 459 and truncated_code <= 519: return 'diseases of the respiratory
            system'; if truncated_code > 519 and truncated_code <= 579:

            return 'diseases of the digestive system'; if truncated_code
        > 579 and truncated_code <= 629:
            return 'diseases of the genitourinary system'; if truncated_code
        > 629 and truncated_code <= 679: return 'complications of pregnancy,
            childbirth, and the puerperium'; if truncated_code > 679 and truncated_code <= 709:

            return 'diseases of the skin and subcutaneous tissue';
        if truncated_code > 709 and truncated_code <= 739:
            return 'diseases of the musculoskeletal system and connective tissue';
        if truncated_code > 739 and truncated_code <= 759:
            return 'congenital anomalies';
        if truncated_code > 759 and truncated_code <= 779:
            return 'certain conditions originating in the perinatal period';
        if truncated_code > 779 and truncated_code <= 799:
            return 'symptoms, signs, and ill-defined conditions';
        if truncated_code > 799 and truncated_code <= 999:
            return 'injury and poisoning'; if 'E' in
    icd9_code:
        return 'external causes of injury';
    if 'V' in icd9_code:
        return 'supplementary classification of factors influencing health status';
```