Intensive Care Unit: Group 1

Aikaterini Baousi, Yuefeng Li, Christos Mourouzi, Akingbolade Shada, Baodong Wang

Abstract
This project focuses in providing a dynamic and accurate way to manipulate, process and visualise the vast amount data in MIMIC III health care database. The data derive from patient admissions in the Intensive Care Unit of Beths Israel Deaconess Medical Centre between June 2001 and October 2012. The report of this project terminates by providing the valuable insights gained by the data manipulation and by mentioning directions for further research. The source code responsible for the loading of the data, the application of Machine Learning algorithms and the visualisation of the data can be found at `https://github.com/ProjectADSICU1/Applied-Data-Science`.

Keywords
Health Data — Applied Data Science—Intensive Care Unit

1. Introduction

The aim of this project is to manipulate and present in a productive way the data stored in a critical care database.To achieve this aim it is necessary to fulfil its objectives. In more detail the objectives of this implementation are to store and preprocess the data, to unveil the underlying dependencies of the data features and make accurate predictions and to create a data-visualisation tool that facilitates the exploration of the data.

The database used in this scope was Medical Information Mart for Intensive Care III (MIMIC) database. This database is ideal for this project because it is sufficiently large and public. It contains information such as demographics, vital sign measurements, laboratory test results, procedures, medications, caregiver notes, imaging reports, and mortality (both in and out of hospital). The tables used in this implementation were the Intesive Care Unit (ICU) Stay Detail and Chart Events. The former contains data pertaining to a unique admission to the ICU such as the gender,date of birth, date of admission to the hospital, date of admission to the ICU, length of stay. The latter contains data on the vital signs that get recorder by clinical staff such as blood pressure, body temperature,heart rate, height, weight.

The workload of this project as well as the outline of this report was split into three basic categories; Data Manipulation, Machine Learning and Visualisation. The first one focused in loading the vast amount of data in MySQL database tables and preprocessing the data. The second category focused in applying Machine Learning algorithms to predict the mortality and length of stay features of the data. The last one concentrated on providing an interactive web-application so that the data would be easily depicted and conclusions about their nature would be effortlessly drawn. A diagram representing the project's architecture is the one shown in Figure 1.

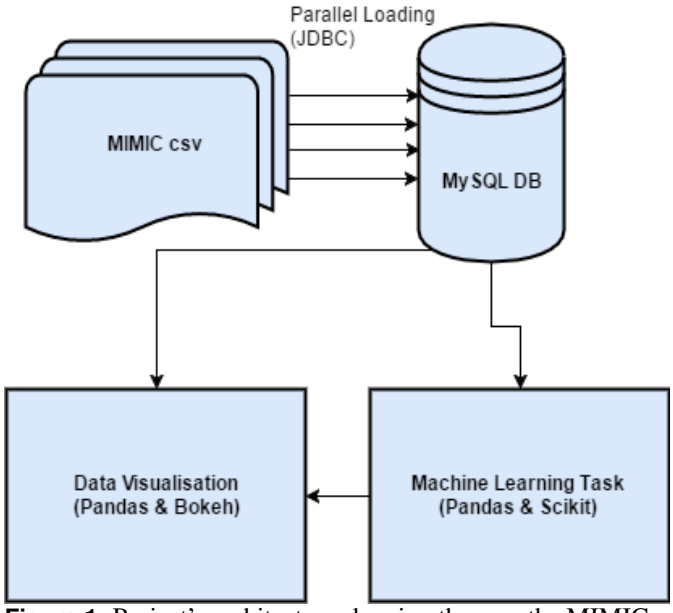

Figure 1. Project's architecture showing the way the MIMIC III data were manipulated and formatted

Conclusively, the realisation of this project was fairly demanding. The enormous amount of data made their manipulation and processing very challenging. Furthermore, a significant amount of them contained frivolous elements that had to be removed so that the results would be accurate and realistic. Nevertheless, the challenges were faced and after a series of evaluation tests (e.g root mean square error, mean absolute error, F_1-score) , we managed to make accurate predictions about critical features of the data and present them in a concise and constructive manner. This provided us with valuable insight and helped us in making suggestions that will improve the patients' hospitalisation in ICU.

2. Related Work

The huge amount of health care information provided by MIMIC database [1] has captured the researchers attention and for that reason there is a significant amount of related, to this project, previous work. Alistair E.W. Johnson et al [2] describe the structure of the MIMIC III database which is an updated version of the widely used MIMIC II database. In their paper, they refer to the development of the database, the patient characteristics, the data records and tables and also the software (available on GitHub [3]) they deployed in order to help researchers to start using the database. M. Saeed et al [4] in their paper describe the use of the previous version of MIMIC to help researchers in intelligent patient monitoring. Numerous researchers have used the MIMIC database to analyse the large dataset so that they can make general or specific conclusions on illnesses or diseases that are recorded in the Intensive Care Unit (ICU). In example, J.X. Sun et al[5] tried to estimate cardiac output from arterial blood pressure waveforms. Also, M.K. Moridanin et al [6] analyse in their paper the heart rate variability as a predictor of mortality in cardiovascular patients of ICU. In the field of data and computer science C. W. Hanson et al [7] reviewed artificial intelligence applications in the ICU and J. Lee et al [8] described a web-based application that could help researches with no computer science background to explore easily the whole dataset of MIMIC II database.

3. Data Description & Manipulation

3.1 Source of the data

The data used in this project were extracted from the MIMIC-III Critical Care Database. MIMIC is a large, freely-available database including health-related data produced by over fifty thousand patients who stayed in critical care units of Beths Israel Deaconess Medical Centre between June 2001 and October 2012. The data inside the MIMIC database are organised into tables. These tables are data storage structures similar to spreadsheets. Each collumn of those spreadsheets contains consistent information and each row contains an instantiating of that information. The MIMIC database structure is shown in Figure 2.

In order to implement the project we had to choose between two options; either to work on a small table containing the derived data (i.e data elements derived from other data elements using a mathematical, logical, or other type of transformation) or to request access to the actual tables containing raw data (i.e unprocessed data). The second option was chosen in order to produce work that is more accurate and realistic.For that reason, it was necessary to pass a CITI "Data or Specimens Only Research" course in order to gain basic knowledge of human research ethics, regulations, laws and local policies prior to initiating research with human subjects. Once this procedure was finished, an application was submitted to PhysioNetWorks and the data were available for download.

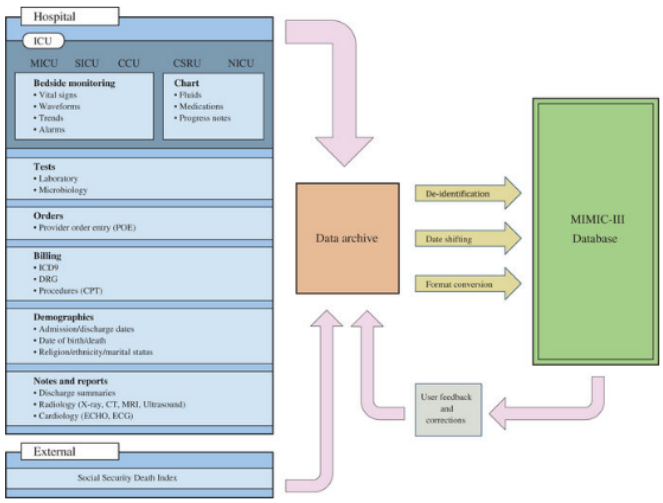

Figure 2. MIMIC database structure

3.2 Data Processing

The initial table downloaded from MIMIC database contained a vast amount of raw data. The main goal at this point of the implementation was to load the data in a MySQL database in an efficient manner and remove orphan data items (i.e records referring to a non-existent patient record) and unrealistic data (e.g 200^oC body temperature). In more detail, data manipulation was deployed in the following way:

- Create the database schema in MySQL

  For this implementation it was necessary to construct the database's schema. Its construction provided an overall database conceptual model that would underlie all the components of the database (e.g tables, views, indexes, stored procedures). Furthermore the schema defines the storage space allocated to host the Intensive Care Unit (ICU) data.

- Create the tables

  The database schema contains the tables of the database. These tables are responsible for storing the produced data inside them. Each of them is mapped to a specific table space and the collection of table spaces belongs to the allocated storage space assigned to the database schema. It is worth mentioning that the tables were row organised.Both the schema and its tables worked as a springboard that helped us in the deeper exploration and understanding of the data.

- Split the initial file into chunks

  The initial CSV file was hard to upload in the schema in a serial manner due to its enormous size. It was necessary to split the initial file into chunks of 10 million rows in order to enable parallel loading.

- Load the chunks in parallel

  The main goal of the parallel loading program was to read each chunk into the memory and then load it to the database's specific table. The code was written in Java and it used Java Database Connectivity (JDBC) application programming interface (API) to connect to the aforementioned MySQL database. The program had to spawn as many threads as possible, considering the hardware resources, to accommodate this workload. Each thread picks a connection from the pool of connections, writes into the database and closes the connection. Writing in this way is more effective than inserting rows one by one because in that way the threads are able to write directly in the memory allocated to the previously mentioned tables. In addition, writing into the table enabled writing directly in the database with respect to the page size limitations. It was possible to use a NoSQL database that implements column organised tables which is proved to be more time saving, storage prudent and faster retriever of the data. However, based on the constrained resources of commodity machines the MySQL procedure seemed more adequate for this implementation. Ultimately, this splitting process was trying to mimic the MapReduce approach of loading the files into the database.

- Facilitate the queries

  Once the loading was completed, the attention was drawn in finding dependencies between the tables produced by the parallel loading of the chunks. Identifying table dependencies was performed by finding the relations between the primary, foreign and composite keys across all the table entries. Therefore, we managed to create indexes that map these dependencies, facilitate and accelerate query execution.

  Once the tables are joined together by indexes and queried, it is necessary to perform aggregative computations to calculate the mean values of continuous variables. Conclusively, it was feasible to get individual rows (i.e rows representing each subjectID) in the training data by pivoting the aggregated mean values.

- Repeat the process

  The process described in the previous steps is repeated in order to extract more information about the medical data (e.g lab results, doctor notes) related to each subjectID.

  Once we have extracted all the valuable information a CSV file was created containing basic information that will help in our project.It should be noted that the initial CSV file that was downloaded from MIMIC database contained 6.000 features for more than 46.000 patients and the final CSV file stored inside the database contained 288 features for 19.000 patients. The aforementioned cleaning procedure helped us produce a file that would be easier to manipulate and extract information from.

4. Machine learning

Once the data were loaded in the database's tables and the impractical values were removed, an attempt was made in order to model the underlying relationships of this dataset and make predictions for future entries. Machine learning techniques can help us extract knowledge from big data, such as those used for this project. The findings focused on predicting the mortality of the patients' entering the ICU and the length of the patients' stay in it, taking into consideration a number of basic features.

4.1 Software

The data manipulation, the application of machine learning algorithms and the evaluation of the performance of these algorithms were implemented using Python programming language.This programming language is a widely used high-level programming language for general-purpose programming.In addition, Python has a vibrant and growing ecosystem of packages.The most related and suitable for the deployment and testing of this project were Scikit-learn [9],Pandas [10] and Matplotlib [11]. The first package has enormous amount of in-built Machine Learning algorithms.In this project this package offered significant help in processing the data, training the models and evaluating the accuracy of the results produced by these models. The second package was used for importing and preprocessing the data (e.g deleting useless entries). Ultimately, the third package was useful in drawing the curves and boxplots depicting the accuracy of the algorithms.

4.2 Preprocessing the data

The following steps were preformed in order to prepare the data before applying the Machine Learning algorithms on them:

- Load the data

  The produced CSV file mentioned in the previous section is loaded using Pandas package. Due to the extensive preprocessing done during the database creation procedure, the loading was straightforward. Nevertheless, it was necessary to once again delete some entries that could have caused problems while performing Machine Learning (e.g delete entries with 0 height and older than 150 years).

- Normalisation

  In order to turn all the data into the same scale we applied normalisation. This helps in introducing the same weight in all of the selected features. Scikit-learn package has in-built functions to facilitate this process.

Dimensionality Reduction	
Pipeline	Accuracy
Feature Selection(5) & Logistic Regression	0,698
Feature Selection(10) & Logistic Regression	0,721
Feature Selection(20) & Logistic Regression	0,722
PCA(5) & Logistic Regression	0,696
PCA(10) & Logistic Regression	0,728
PCA(20) & Logistic Regression	0,737

Table 1. Table presenting the accuracy of Logistic Regression with 5, 10, 20 features after applying feature selection and feature extraction.

- Split the data

  75% of the data were used for training purposes and 25% were used for testing. This ratio was selected in order for the training data sample to have a lower variance. In addition, this way of splitting the data helps us prevent overfitting. If all the data were used to train the model, that would have caused the model to be rigid and hard to correspond to new data. This splitting provided a frame of reference and enabled the conduction of rigorous experiments (e.g mortality predictions, length of stay predictions).

4.3 Mortality Predictions

Once the steps mentioned in the previous section were completed, the Machine Learning algorithms could use the underlying information in the data to make predictions regarding the patients', entering the ICU, mortality. In our implementation we wanted to achieve the highest accuracy rate since this is a very crucial feature. Therefore it was necessary to explore how the Machine Learning algorithms would behave after applying dimensionality reduction or taking into consideration all the features of our data.

The dimensionality reduction approach can be divided in two subcategories; feature selection and feature extraction. The feature selection we applied for this project was based on chi-square test , which focuses on evaluating the dependencies between the different features of the data and selecting the most dependent features.This approach was combined with Logistic regression and managed to achieve 0.722 accuracy for 20 selected features. We also followed the feature extraction approach using Principal Component Analysis (PCA) which "summarises" the data in a number of features.Once this procedure managed to reduce the dimensionality of the dataset, Logistic Regression was applied again. In the optimal case , the dimensionality of the data was reduced in 20 features and the accuracy of the model was 0.737. These results and the results after reducing the dimension of the data into 5 and 10 features are shown in the table 1.

In search of better accuracy we tried to take into consideration all 288 features. The classification algorithms tested for that purpose are Logistic Regression (LR), Linear Discriminant Analysis (LDA),k-Nearest Neighbours (KNN), Classification And Regression Tree (CART), Naive Bayes (NB) and linear Support Vector Machine (SVM). The two algorithms that performed the best were LR and LDA with accuracy values 0,751 and 0,747 respectively. In more detail the accuracy values of all the classification algorithms tested in this implementation are shown in Figure 3.

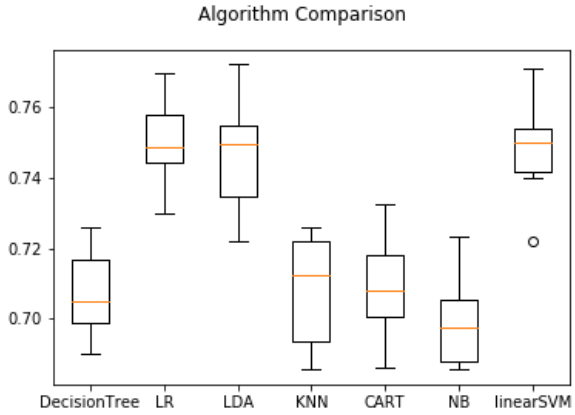

Figure 3. Representation of the variation of the accuracy values of the classification models.

The accuracy values obtained in this procedure seemed very promising but it was also necessary to evaluate the precision of these results as well. For that reason, values such as the precision, recall and F_1 score of the predictions had to be calculated. In order to realise the meaning of the previous variables it is crucial to understand the terms true positive, true negative, false positive and false negative. This can be achieved with the following definitions:

- True Positive (TP): Number of correct matches.
- True Negative (TN): Non-matches that were correctly rejected.
- False Positive (FP): Proposed matches that were incorrect.
- False Negative(FN): Matches that were not correctly detected.

Recall is the number of true positives divided by the total number of elements that actually belong to the positive ($Recall = \frac{TP}{TP+FN}$). It provides a measure to estimate the integrity of the results. In the following plot the recall of the results produced by the Machine Learning algorithms mentioned in this section is visualised:

Precision is defined as the number of true positives divided by the total number of elements labelled as belonging to the positive class ($Precision = \frac{TP}{TP+FP}$). It provides a measure

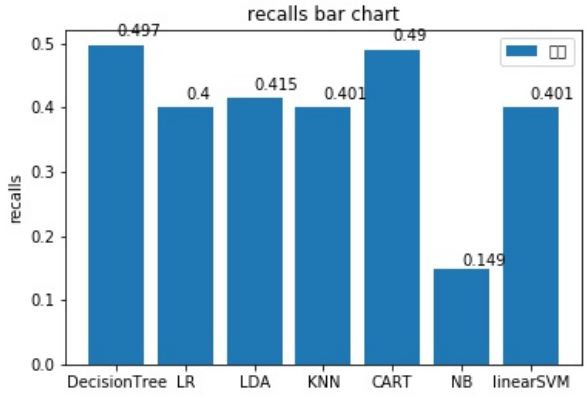

Figure 4. Recall results that quantify the integrity of the results.

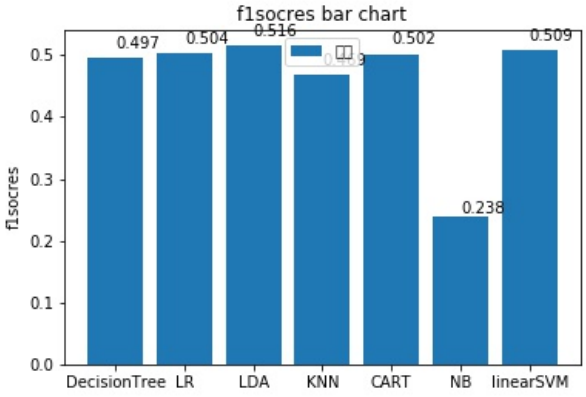

Figure 6. F_1-scores of the results produced in every case denoting the practicality and importance of the estimations.

to estimate the practicality of the results. In Figure 5 the precision of the results produced by the Machine Learning algorithms mentioned in this section is visualised.

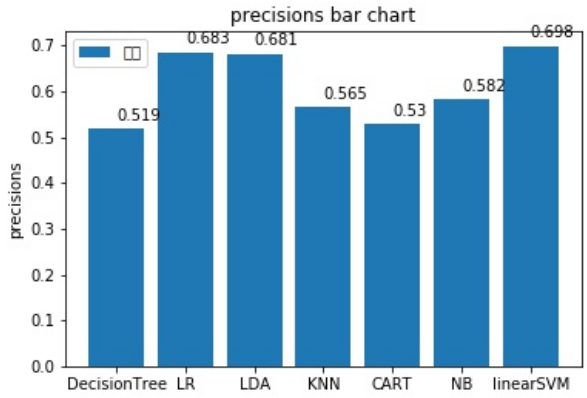

Figure 5. Precision results that quantify the practicality of the results.

The measure that combines the precision and recall is the harmonic mean of precision and recall, the F_1-score. In other words, the F_1-score evaluates the completeness and usefulness of the results produced by the Machine Learning algorithms. It is evident in Figure 6 of the F_1-score that the models with the highest values are the ones that provided the highest accuracy, making our results and deductions trustworthy.

Another way to obtain information concerning the performance of the classifiers is by plotting their ROC-curves (TP vs. FP). In this way, the performance of the classifier over its entire operating range is assessed.The most widely-used measure is the area under the curve (AUC). In Figure 7 we can see that the models that preformed the best (i.e had the highest AUC values) were the ones that provided the highest accuracy values.

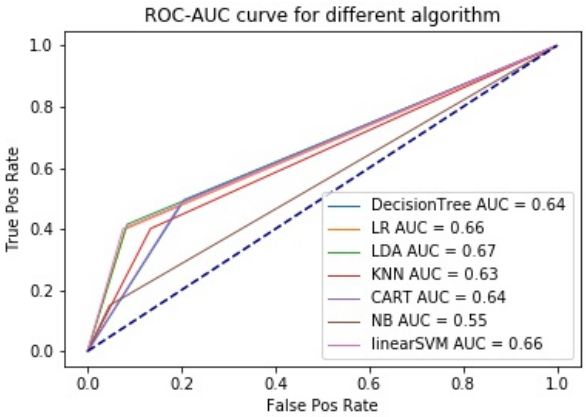

Figure 7. ROC curves of the classifiers used in this implementation.

4.4 Duration of Hospitalisation Predictions

An attempt was made in order to predict the length of stay in ICU based on a number of features.The same initial procedure was followed (i.e loading, normalising and splitting the data) and then Decision Tree Regression (DTR) and Logistic Regression (LR) algorithms were used. Both these algorithms provide us with models that can help us estimate the duration of a patient's hospitalisation in ICU.

In order to evaluate the precision of the results the root mean square error metric and the mean absolute error metric were calculated.Both of them are commonly used error metrics used in measuring the performance of regression models by quantifying the "distance" between the estimated values and the true value. The former is highly influenced by the extreme values of the dataset and the latter assigns equal weight to the spread of data. Therefore both of them should be taken into consideration in order to produce accurate deductions. In the following tables and figures we present the variations in the values of the error metrics for both regression algorithms as the number of features increases.

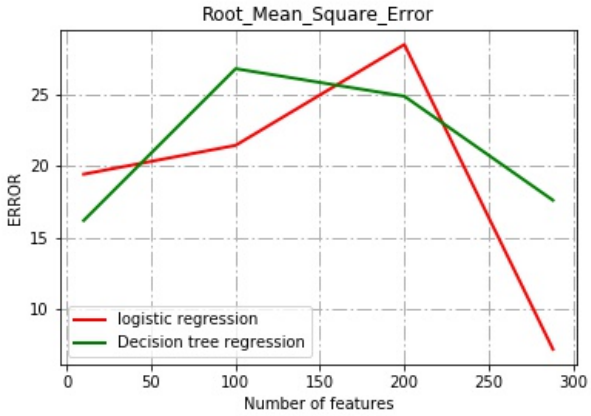

Figure 8. Root Mean Square Error with evolving number of features.

Root Mean Square Error		
Features	LR	DTR
10 Features	19,407	16,177
100 Features	21,418	26,768
200 Features	28,455	24,845
288(all) Features	7,192	17,591

Table 2. Root Mean Square Error for 10, 100, 200 and all features.

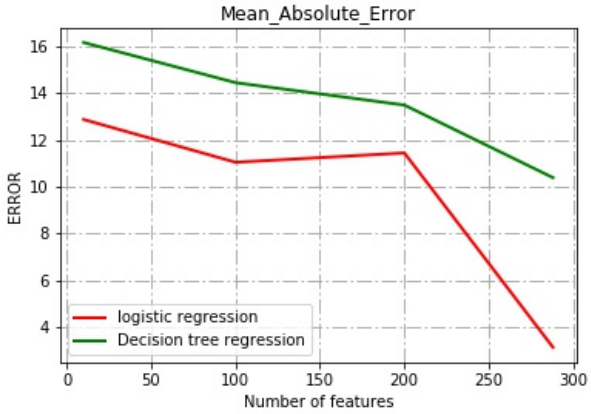

Figure 9. Mean Absolute Error with evolving number of features.

Mean Absolute Error		
Features	LR	DTR
10 Features	12,881	16,177
100 Features	11,051	14,458
200 Features	11,450	13,500
288(all) Features	3,135	10,394

Table 3. Mean Absolute Error for 10, 100, 200 and all features.

It is noticeable in these tables(2, 3) and figures (8, 9) that the best results derived from the Logistic Regression algorithm which provided a model that predicted the duration of hospitalisation with mean absolute error of 3 days. The Decision Tree Regression algorithm gives a lot more inaccurate results since its mean absolute error is 10.39 days. The root mean square error of both algorithms is very high which signifies that the dataset contains many extreme values.

4.5 Deductions

The Machine Learning results provided useful insight to the data. These results can turn out to be very helpful for the administration of the hospital. By being able to predict the mortality of a newly submitted patient and provide accurate results 75% of the times, it can be possible to prioritise people in critical condition over other patients in a less dangerous position. Furthermore, it is possible to estimate the length of stay in ICU and therefore the hospital can increase the number of beds in the halls or doctor shifts when many patients are going to be hospitalised for a long time.

5. Visualisation

A data visualisation tool was implemented in order to enable a quick, convenient and interactive exploration of the processed by the previous steps data. This tool could aid researchers in discovering the distinctive characteristics of the data and therefore facilitate their preliminary investigation. Furthermore, this implementation provides significant help since it prevents the interaction with raw data and presents the information in a more sophisticated manner. In this way, patterns can be easily detected and reasonable conclusions could be drawn.

5.1 Software

The visualisation tool was deployed in Python programming language. Once again the numerous programming packages of this language facilitated the realisation of this project. The libraries used in this section were Bokeh and Pandas which were responsible for the data presentation and the data analysis respectively [12, 10]. Bokeh provides an interactive, elegant and concise construction of novel graphics. Its main goal is to extend the visualisation experience with high-performance interactivity over very large datasets. For that reason this package is the most suitable for this particular implementation. In addition, as mentioned in Machine Learning software section, Pandas library simplified the data-manipulation and information extraction procedures.

5.2 Web-application

The implemented visualisation tool was designed to provide two main features; exploration and comparison. This tool involves a dynamic 2D plot representing patients as dots. The variables represented by the x-axis and y-axis can be interactively selected by the user giving the opportunity to depict as many data characteristics as possible. These variables consist

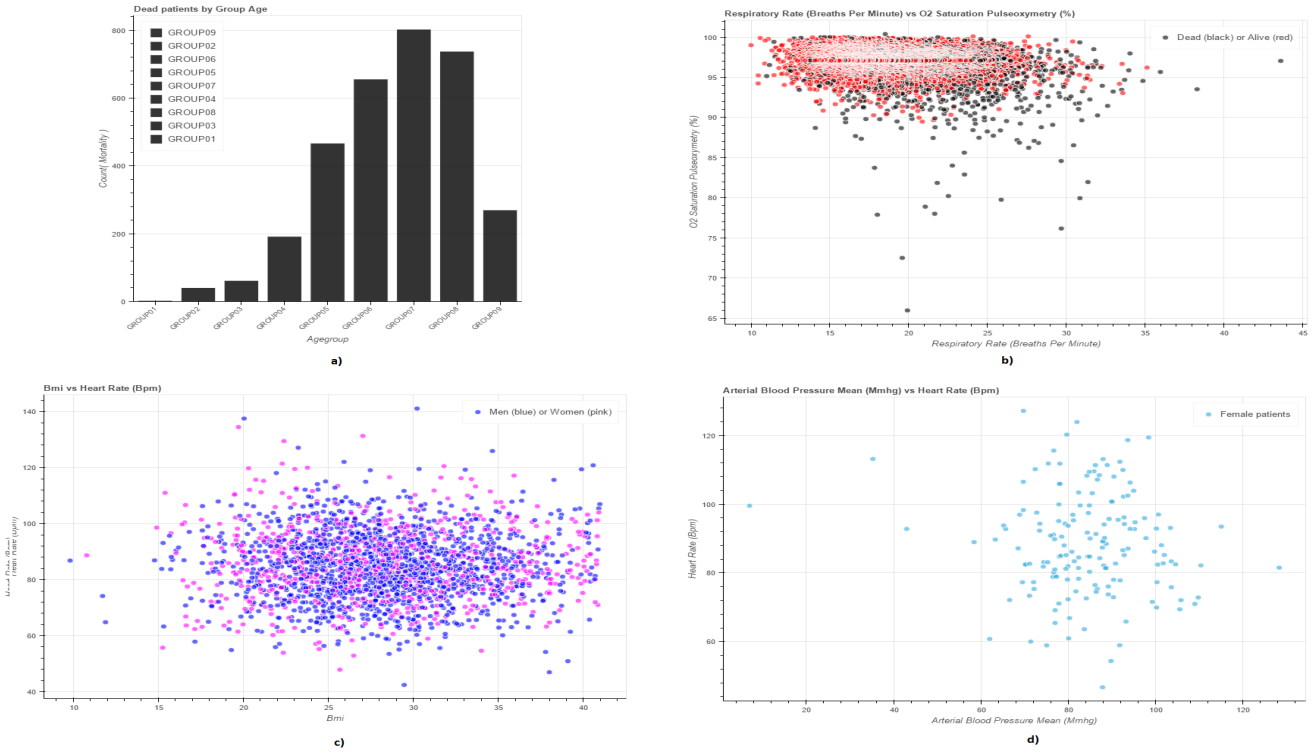

Figure 10. Produced plots by the web application.

of vital sign measurements taken during each patient's hospitalisation in ICU. Additionally, the information presented in these graphs can be represented in specific cohorts such as mortality (i.e dead or alive patients), gender (i.e female or male patients) and age groups (i.e patients from the age of 10 until the age of +89 in groups that span in 10 years). The comparison between the data entities or subsets of the data entities is facilitated by the colour selection toolbar of the application. In this way the user can colour each patient dot according to a specific feature (e.g dead-black/alive-red).

The dimensionality of the data can be additionally reduced by producing aggregate variables (i.e variables that derive from several features). The derived variable we used in this implementation is the Body Mass Index (BMI) that is defined as the body mass divided by the square of the body height, and is universally expressed in units of kg/m^2. The BMI attempts to quantify the amount of tissue mass (muscle, fat, and bone) in an individual, and then categorise that person as underweight, normal weight, overweight, or obese based on that value. This value is really significant in the scope of our implementation because it combines the information provided by two features (i.e height and weight) into one. Therefore it is easier to draw conclusions on the way a patient's physical condition affects his/hers vital measurements (e.g heart rate, blood pressure).

The application is structured according to the following steps:

- Load the data: The data are retrieved from MySQL database and saved in a Pandas' dataframe.
- Initialise the figure: The control box (i.e selection bars and radio buttons) is created and an initial figure is constructed.
- Update the figure: Every time the user chooses a different attribute from the control box the figure is reloaded and updated.A new plot is created presenting the user's options.

We also created a set of bar charts and histograms that show the distribution of the admitted to ICU patients according to a feature.

5.3 Examples

In Figure 10 there is a set of possible plots produced by this application. In those figures we can clearly see the way this implementation enables the comparison and exploration of the data. In more detail if we examine the top-left plot (a) we can see a bar chart representing the number of deceased patients per age group. The top-right figure (b) is a dot plot representing the alive data entities in red colour and the deceased data entities in black colour. The y-axis of this graph shows the levels of oxygen saturation (%) in the patients' blood and the x-axis shows the respiratory rate (breaths per minute) of each patient. The bottom-left figure (c) is a dot plot representing the male data entities as blue dots and the female data entities as pink dots in the age group spanning from 60 to 69 years. The plot visualises each patient's, in this age group, BMI (x-axis) and heart rate (y-axis). The bottom-right figure (d) is a

dot plot that displays the alive female data entries in the age group spanning from 40 to 49 years. The x-axis portrays the mean arterial blood pressure (mmHg) and the y-axis the heart rate (bpm).

6. Insights

The aforementioned procedures gave us the opportunity to gain valuable insights to the data. The Machine Learning algorithms helped us discover the veiled data dependencies, train models and make accurate predictions. The implemented web-application promoted the meaningful visualisation of the data and helped us understand them thoroughly. The insights gained by these two implementations are explained in more detail below.

- **Machine Learning Insights**

  The Machine Learning algorithms focused in predicting 2 main features of the dataset: the mortality and length of stay in ICU.The algorithms used for these purposes performed well and provided accurate results that further the extraction of significant outcomes.

  After putting the dataset to the test of numerous Machine Learning algorithms we managed to find the algorithm that both performed the best (F_1-score=0,504) and provided the most accurate predictions (accuracy=0,771) regarding the mortality feature.The accuracy plays a more important role in these predictions since the feature that has to be predicted is very crucial.The Logistic Regression algorithm can predict correctly 75% of the times the mortality status of the patient. This insight is extremely valuable since it can help the hospital staff identify accurately and realistically whether a newly submitted patient, in the ICU, is in a critical condition or not based on his/hers initial vital measurements. This will help the hospital staff prioritise some patients that are more likely to be in danger than others that are in a more stable condition and therefore decrease the mortality rate of the patients in the ICU.

  The Logistic Regression algorithm provided the best results in the prediction of the duration of hospitalisation in the ICU as well. This algorithm returns a real number that represents the possible number of days a patient can stay in the ICU with mean absolute error of 3 days. This insight is really helpful as well because it can help the hospital's administration calculate the workload in ICU based on the newly submitted patients' vital measurements. In this way the hospital can be prepared to tackle with many simultaneous hospitalisations and provide solutions such as increase the number of hospital bed's and medical shifts in ICU.

- **Visualisation Insights**

  The web-application created as part of this project focused on providing a meaningful and interactive tool that would represent the data of the vital measurements for each admission in the ICU.Some of the most interesting insights produced by this tool are presented below.

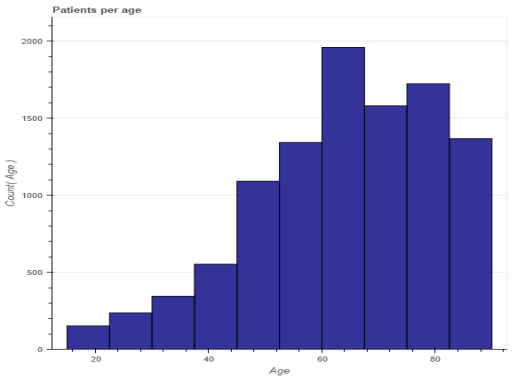

Figure 11. Histogram presenting the distribution of admissions per age group.

This histogram shows the distribution of the admitted patients in each age group. The age range is divided in 10 bins. It is evident from the histogram that the age of the patients most frequently admitted in the ICU spans from 60 until 67 years. The admissions of patients ageing more than 60 years old are significantly more than the ones occurring in a younger age. This is an expected result since this age group faces more health issues.

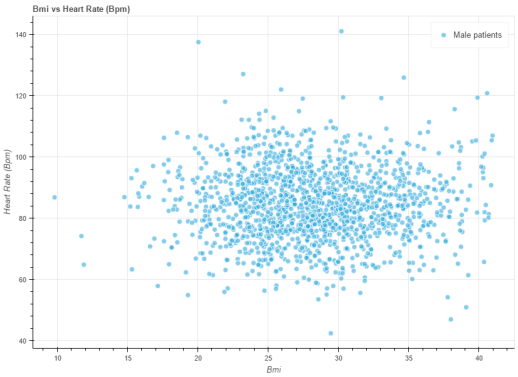

Figure 12. BMI vs Heart Rate male patients aged 60-69.

Figures 12,13 represent respectively the BMI vs heart rate of both the male and female patients aged between 60 and 69 years which is the most populated age group according to the previous histogram. The female entries spread more than the male entries since their BMI and heart rate measurements exhibit variance. The majority of the male patients' measurements tend to have heart rate between 68 bpm and 96 bpm and their BMI index ranges between 20 kg/m^2 and 35 kg/m^2. The same observations hold for female patients but a conclusion cannot be easily drawn in this case due to the low clustering of the data.

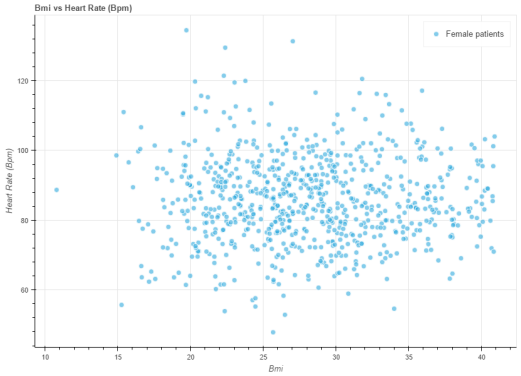

Figure 13. BMI vs Heart Rate female patients aged 60-69.

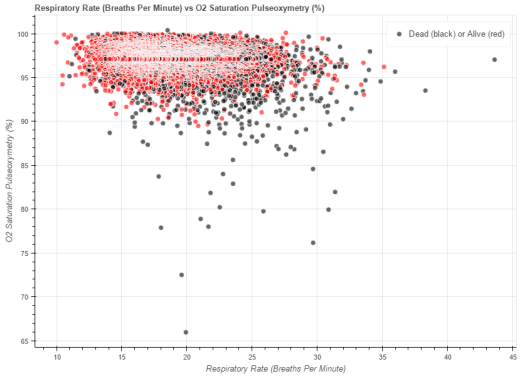

Figure 14. Oxygen Saturation vs Respiratory rate of alive (red dots) and deceased (black dots) patients.

Figure 14 depicts the oxygen saturation(%) and the respiratory rate (breaths per minute) of the living (red dots) and deceased (black dots) patients. In the plot it is noticeable that the patients with oxygen saturation below 93,5% enter a critical stage. Patients exhibiting lower than 96% saturation rate start to increase their respiratory rate(20 breaths per minute and more) in order to receive more oxygen.

7. Conclusion

The project aimed at manipulating the raw data provided by the MIMIC III healthcare database. Initially the data were split and then stored in parallel inside a MySQL database. Then the senseless values of the data were removed and we proceeded to the extraction of information from the hidden dependencies in them. In order to make accurate predictions of some crucial data features such as the mortality and length of stay in ICU we applied Machine Learning algorithms. These algorithms gave us the ability to predict the mortality with 75% accuracy and the length of stay with mean absolute error of 3 days. Ultimately, the visualisation tool provided a set of plot that can be dynamically modified in order to serve the users needs create multiple views of the data.

Some future implementations, that can broaden the use of this project, are the creation of a mobile application and the expansion of set of features by researchers.The mobile application can ease the storage of patients' vital measurements in the database and produce more robust and real-time results.The massive database that has been used in this project can be stored in the clouds in order to solve the problems of size and time. In this way the application would be able to access the data with ease and store new values representing new patient admissions. With this mobile application the hospital staff will be able to access the MIMIC III database in the cloud and benefit from the faster data processing in order to make real-time deductions. For example, they can store the vital measurements of a newly admitted patient and retrieve the information produced by the Machine Learning algorithms directly.Furthermore, researchers can profit from the findings of this project since they will be able to focus on the most interesting features of the data set and try to improve them in order to produce more precise results. The researchers can also expand the features of the database by including the lab results that will help in providing solutions in cases of emergency.

References

[1] The scope of mimic. `http://criticaldata.mit.edu/scope-of-mimic/`. Published: 04/08/2014.

[2] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3, 2016.

[3] Website & documentation for the mimic critical care database. `https://github.com/MIT-LCP/mimic-website`.

[4] Mohammed Saeed, Christine Lieu, Greg Raber, and Roger G Mark. Mimic ii: a massive temporal icu patient database to support research in intelligent patient monitoring. In *Computers in Cardiology, 2002*, pages 641–644. IEEE, 2002.

[5] JX Sun, AT Reisner, M Saeed, and RG Mark. Estimating cardiac output from arterial blood pressurewaveforms: a critical evaluation using the mimic ii database. In *Computers in Cardiology, 2005*, pages 295–298. IEEE, 2005.

[6] Mohammad Karimi Moridani, Seyed Kamaledin Setarehdan, Ali Motie Nasrabadi, and Esmaeil Hajinasrollah. Analysis of heart rate variability as a predictor of mortality in cardiovascular patients of intensive care unit. *Biocybernetics and Biomedical Engineering*, 35(4):217–226, 2015.

[7] C William Hanson III and Bryan E Marshall. Artificial intelligence applications in the intensive care unit. *Critical care medicine*, 29(2):427–435, 2001.

[8] Joon Lee, Evan Ribey, and James R Wallace. A web-based data visualization tool for the mimic-ii database. *BMC medical informatics and decision making*, 16(1):15, 2016.

[9] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[10] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[11] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[12] Bokeh Development Team. *Bokeh: Python library for interactive visualization*, 2014.