Author/s:
Sefrioui, I;Amadini, R;Mauro, J;El Fallahi, A;Gabbrielli, M

Title:
Survival prediction of trauma patients: a study on US National Trauma Data Bank

Date:
2017-12-01

Citation:
Sefrioui, I., Amadini, R., Mauro, J., El Fallahi, A. & Gabbrielli, M. (2017). Survival prediction of trauma patients: a study on US National Trauma Data Bank. European Journal of Trauma and Emergency Surgery, 43 (6), pp.805-822. https://doi.org/10.1007/s00068-016-0757-3.

Persistent Link:
https://hdl.handle.net/11343/282668

ORIGINAL RESEARCH

Survival Prediction of Trauma Patients: A Study on US National Trauma Data Bank

Imane Sefrioui[1*], Amadini Roberto[2], Jacopo Mauro[3], Abdellah El Fallahi[4] and Maurizio Gabbrielli[5]

Abstract

Background: Exceptional circumstances like major incidents or natural disasters may cause a huge number of victims that might not be immediately and simultaneously saved. In these cases it is important to define priorities avoiding to waste time and resources for not savable victims. Trauma and Injury Severity Score (TRISS) methodology is the well-known and standard system usually used by practitioners to predict the survival probability of trauma patients. However, practitioners have noted that the accuracy of TRISS predictions is unacceptable especially for severely injured patients. Thus, alternative methods should be proposed.

Methods: In this work we evaluate different approaches for predicting whether a patient will survive or not according to simple and easy measurable observations. We conducted a rigorous, comparative study based on the most important prediction techniques by using real clinical data of the US National Trauma Data Bank.

Results: Empirical results show that well-known Machine Learning classifiers can outperform the TRISS methodology. Based on our findings, we can say that the best approach we evaluated is Random Forest: it has the best accuracy, the best Area Under the Curve, and k-statistic, as well as the second-best sensitivity and specificity. It has also a good calibration curve. Furthermore, its performance monotonically increases as the dataset size grows, meaning that it can be very effective to exploit incoming knowledge. Considering the whole dataset, it is always better than TRISS. Finally, we implemented a new tool to compute the survival of victims. This will help medical practitioners to obtain a better accuracy than the TRISS tools.

Conclusion: Random Forests may be a good candidate solution for improving the predictions on survival upon the standard TRISS methodology.

Keywords: Survival Prediction; Trauma Patients; National Trauma Data Bank; Classification; Machine Learning; TRISS methodology

1 Introduction

Exceptional circumstances like major incidents or natural disasters may result in a huge number of victims that might not be immediately and simultaneously saved, e.g., due to limited resources. In these cases it is important to define the priorities. In particular, it would be detrimental to waste time and resources trying to save not savable victims.

The *triage* is the process of determining the priority of patients' treatments based on the severity of their condition. Usually, a primary triage is carried out at the scene of an accident, while a secondary triage is performed at a *Casualty Clearing Station* (CCS), often located between the inner and outer cordons of the major incident. Several triage protocols exist for handling emergencies. One of the most widely used is the *START* protocol [1], which is adopted in the United States, Canada, Saudi Arabia, and Australia. Patients are assigned to four categories:

- *red* (top priority) for those requiring immediate priority and an immediate response;
- *yellow* (delayed priority) for patients needing urgent medical attention, but can be delayed for a short time (i.e., should be treated within 20 minutes);
- *green* (minor priority) for patients that are conscious, breathing and have minor injuries (i.e., should be treated within 1 hour);
- *black* (low priority) for patients that are dead or impossible to save.

*Correspondence: sefrioui.imane@gmail.com
[1] Faculty of Sciences of Tetouan, University Abdelmalek Essaadi
Full list of author information is available at the end of the article

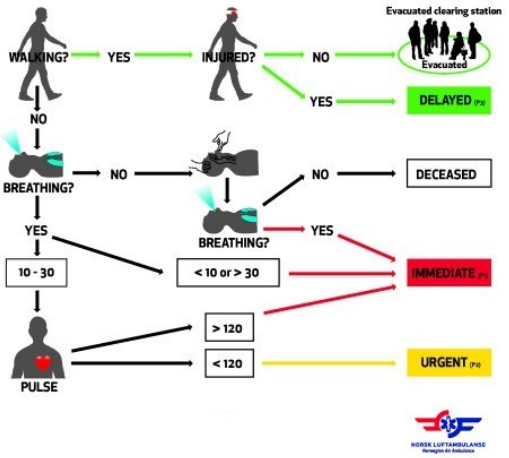


Figure 1: Triage Sieve Protocol (Norwegian Air Ambulance).

Another triage protocol, adopted in part of Europe and accepted by NATO, is the *Triage Sieve* [2] depicted in Figure 1 [3]. Similarly to START protocol, also the Triage Sieve schema classifies the victims into four categories according to the patients' conditions.

When more resources are available, the patients will undergo a further, more detailed triage based on vital signs. In particular, many different *trauma scoring* systems have been developed for measuring the seriousness of the patients and for scheduling the relief efforts. For example, some are based on physiological scores (e.g., the Glasgow Coma Scale [4]) while others rely on anatomical description (e.g., the Abbreviated Injury Scale [4]). There is no universally accepted scoring system and each system has its own limitations. In general, only physiological features such as blood pressure and respiration rate are assigned at first contact, while precise determination of anatomical damage is usually not possible at the scene of injury. Injury severity can be assessed or estimated by medical services physicians in the prehospital phase [5, 6] but the assessment of injury severity by emergency physicians based on physical examination at the scene of injury is not always reliable [7, 8].

In this paper we first evaluated different statistical methods for tackling a major aspect of these more detailed triages that focus on predicting if a patient will survive or not according to simple and easy measurable observations. We conducted a rigorous, comparative study based on some of the most important Machine Learning (ML) techniques. We validated our experiments by using the information stored in the US *National Trauma Data Bank* (NTDB), a database containing over 2.7 million records of patients hospitalized with traumatic injuries in 2002, 2003, 2004, and 2006. Since in our case the prediction task is a binary classification problem (i.e., the output of a survival prediction is either "dead" or "alive") we evaluated some of the best-known Machine Learning classifiers including Artificial Neural Networks, Naive Bayesian Classifier, Support Vector Machine, and Decision trees. We compared the obtained results with the *Trauma and Injury Severity Score* (TRISS) methodology [9], the standard model typically used for predicting the survival probability of trauma patients. Empirical results show that the accuracy of standard Machine Learning classifiers can be better than the TRISS methodology. In particular, the most accurate method was found to be Random Forest [10], a ML approach based on decision trees.

According to the results of the above empirical evaluation, we then developed the second major contribution of this paper: a C++ tool for survival prediction based on the best ML classifiers we tested, including Random Forest and other approaches. The tool takes as input the patient information (e.g., blood pressure or respiratory rate) and estimates whether the patient will survive or not.[1] We hope that this tool can help practitioners to predict the survival of patients with a better accuracy than TRISS-based tools (e.g., [11, 12]).

Paper structure. In Section 2 we give some preliminary notions about the TRISS approach, the NTDB database, and the classifiers used in the experiments. Section 3 describes the methodology we followed to conduct the experiments, while in Section 4 we show the obtained results. In Section 5 we present the tool we developed to perform the survival predictions. In Section 6 we report the related work before concluding in Section 7.

2 Background

In this section we recapitulate some preliminary notions about the TRISS approach, the NTDB database, and the classifiers we evaluated in our experiments.

2.1 TRISS model

The Trauma and Injury Severity Score (TRISS) method [9, 13, 4] is a standard and well-known model typically used for predicting the survival probability of trauma patients. It is based on logistic regression [14] with four predictors: the age of the patient, the type of injury (either blunt or penetrating), the *Revised Trauma Score* (RTS), and the *Injury Severity Score* (ISS). RTS and ISS result from a combination of other parameters. In particular the RTS is the weighted sum of

[1] The tool is available online at https://github.com/imanesefrioui/tool_prediction_survival_executable

the blood pressure, the respiratory rate, and the *Glasgow Coma Scale* (GCS) [4], a neurological scale that aims to give a reliable, objective way of recording the conscious state of a person. The ISS provides an overall score for injury severity in 6 body regions (head, face, neck, abdomen, extremities, external) according to the *Abbreviated Injury Scale* (AIS) [4]. The severity of the injuries ranges from 0 (minor injury) to 6 (not survivable injury). The ISS is the sum of the squares of the highest injury severity scores.

TRISS determines the probability $\mathbb{P}$ of survival for a patient X via the following logistic regression function:

$$\mathbb{P}[X \text{ survives}] = \frac{1}{1+e^{-b}}$$

where:

$$b = \begin{cases} b_0 + b_1 \text{ RTS} + b_2 \text{ ISS} & \text{if } X \text{ is less than} \\ & 55 \text{ years old} \\ b_0 + b_1 \text{ RTS} + b_2 \text{ ISS} + b_3 & \text{otherwise} \end{cases}$$

and b_0, b_1, b_2 and b_3 are the *regression parameters*.

The regression parameters were derived from multiple regression analysis of the *Major Trauma Outcome Study* (MTOS) database in the 1990s. The MTOS until recently served as a standard reference database of seriously injured patients in the United States, and was the basis for many of the analytic methods that have become nowadays familiar. For blunt injuries $b_0 = -0.4499$, $b_1 = 0.8085$, $b_2 = -0.0835$, $b_3 = -1.7430$, while for penetrating injuries $b_0 = -2.5355$, $b_1 = 0.9934$, $b_2 = -0.0651$, $b_3 = -1.1360$.

Different software tools for automatically computing the TRISS value starting from the patient input data have been developed and made available on-line [11, 12].

We would like to remind that TRISS relies on the AIS which is seldom calculated upon hospital admission, as it often requires primary, secondary, and tertiary patient surveys to assess all injuries completely. Thus, TRISS is rarely used for early baseline risk adjustment.

2.2 National Trauma Data Bank

Starting from 1982 the *American College of Surgeons Committee on Trauma* (ACS COT) coordinated the MTOS. At the conclusion of MTOS, the ACS COT renewed its commitment to trauma research and quality improvement by developing trauma registry software. In 1997 a subcommittee was established to direct the *National Trauma Data Bank* (NTDB) database. The NTDB is the largest aggregation of US trauma registry data ever assembled. Currently, the NTDB contains detailed data on over 2.7 million cases from over 900 US trauma centres. These data have been shared with hundreds of researchers and represented the basis of numerous scientific publications [15, 16].

The NTDB consists of data submitted by participating hospitals, and is continually cleaned and standardised to improve data quality and consistency. It is therefore one of the best sources of information to test trauma survival prediction models. Each record of NTDB represents the vital signs of a given patient, i.e., physiological scores, anatomical descriptions, and physical measures. More technicalities about NTDB are given in Section 3.1.

2.3 Machine Learning and Classification

Machine Learning (ML) is a broad field that uses concepts from Computer Science, Mathematics, Statistics, Information Theory, Complexity theory, Biology and Cognitive Sciences to *"construct computer programs that automatically improve with experience"* [17]. In this paper we are interested in *classification*, a well-known ML problem that consists in identifying into which categories or classes a new observation belongs by means of appropriate classifiers. A classifier is therefore a function that maps a new, unseen problem instance characterized by a collection of numerical features to one of a finite number of classes on the basis of a training set of instances whose class is already known [17]. In particular, here we are interested in *binary classification*: the prediction output for each incoming patient is a binary value that can be either "dead" or "alive". In order to assess what are the best practices in this context we tested different well-known classifiers, that we briefly describe in the rest of the section.

Artificial Neural Networks

Artificial Neural Networks (ANN) are a family of learning models inspired by the way the brain and the nervous system work. They are used to estimate or to approximate functions that can depend on a large number of inputs. They are formed by a systems of interconnected "neurons" which can compute values from inputs producing output based on suitable weights. Training an ANN means learning the weights that regulate the output of the neurons. Different approaches have been proposed in the literature [18].

In this paper we consider one of the simplest ANN training methods: the *back propagation*. Weights are initially random values. Then they are adjusted by applying a gradient descent method based on the output expected on some instances used to train the network. In medical research, the most commonly used artificial

neural networks are the *Multilayer Perceptrons* (MLPs) with backpropagation [19].[2]

Decision Trees

Decision Trees (DTs) techniques rely on trees (or forest of trees) for making predictions. The tree structure basically consists of leaves, representing the class labels, and branches, representing conjunctions of features. Decision Trees and their extensions are widely adopted in classification since they can lead to very effective performance.

Here in particular we consider two relevant approaches based on DTs: Random Forest and C4.5. Random Forest constructs a forest of random trees, while C4.5 is a method based on the normalized information gain ratio. Partial C4.5 DTs are also used by other ML approaches, for example rule-based methods relying on PART decision lists. For more detailed explanations about these techniques, please see [10, 20, 21].

k-Nearest Neighbours

The k-Nearest Neighbours (k-NN) algorithm is a non-parametric method used for classification and regression [22]. Given a notion of distance (e.g., Euclidean or Manhattan distance) the prediction is performed according to the $k > 0$ instances closer to the instance to classify. In classification, a new instance is typically classified by a majority vote of its neighbours. The choice of k is critical and typically application-dependent: generally, larger values of k reduce the effect of noise but make boundaries between classes less distinct.

The k-NN algorithm is among the simplest of all ML algorithms. Its underlying assumption is that "similar instances have similar behaviours". It is also referred as a *lazy* approach, since the learning function is only approximated locally and all computation is deferred until classification.

Logistic Regression

Logistic regression (or logit regression, logit model) is a direct probability model developed by D.R. Cox in 1958 [14]. It uses a logistic function for modelling the probabilities of the possible outcomes. The logit model is a generalised linear model dependent on different parameters. Different methods can be used to estimate the model parameters, e.g., the maximum likelihood estimation or the minimum chi-squared estimator for grouped data. Penalized maximum likelihood estimation with a quadratic penalty function is often called *ridge* logistic regression [23].

Logistic regression is commonly used in Machine Learning and applied in many fields, including medical and social sciences. For example, the aforementioned TRISS and many other medical methods for assessing the severity of a patient rely on logistic regression [24, 25, 26, 27].

Naive Bayes

The Naive Bayes classifier provides a simple approach for representing, using, and learning probabilistic knowledge. One can view such a classifier as a specialised form of Bayesian Network. The term *naive* comes from two important simplifications: it assumes that the features are conditionally independent given the class, and that no hidden feature (i.e., a variable that is not observable) influences the prediction process. A common assumption, not intrinsic to the Naive Bayes approach but often made nevertheless, is that within each class the values of numeric attributes are normally distributed. These assumptions allows Naive Bayes to be very efficient from the computational point of view. For more information on Naive Bayes classifiers, we refer the interested reader to [28].

Support Vector Machines

Support Vector Machines (SVMs) are techniques that aims at finding the hyperplane that partitions a dataset into two disjoint subsets. The idea is to consider the hyperplane that allows the samples of each class to stay on opposite sides of the separating plane. When there is no hyperplane that can separate the positive instances from the negative ones, SVMs map the data to a higher dimensional space that allows to define a separating hyperplane. To do so, some popular kernels such as polynomial and Radial Basis Function are used.

The *Sequential Minimal Optimization* (SMO) algorithm is a method devised by John Platt for solving the Quadratic Programming problem that arises during the training of support vector machines. SMO is widely implemented and used for training SVMs. For more details on SVM and SMO, see [29, 30, 31].

3 Methodology

Taking as baseline the best practices for testing classification techniques [32, 33], in this section we present the main ingredients and procedure that we used for conducting our experiments and evaluating the classifiers.

[2] In our case the learning rate was set to 0.3, the momentum to 0.2, the number of iterations to 500, and the hidden layer composed of 9 nodes.

3.1 Dataset

In this work we consider the US National Trauma Data Bank (NTDB) introduced in Section 2.2. We used version 7.2 of NTDB, containing the records of almost 2 millions patients hospitalised with traumatic injuries in 2002, 2003, 2004, and 2006. We focused only on trauma patients, i.e., those having the field injury type set to 'blunt' or 'penetrating'. For the sake of consistency, as done in numerous studies and as suggested in the database manual [34], we discarded all the records having missing information or out-of-range values.

Following the methodology of [15], every patient is characterised by 17 different features, 15 numerical and 2 nominal (viz., Gender and Injury Type). The list of these features is shown in Table 1.

Table 1: Features of a patient in the NTDB database.

	Name	Range
x_1	Age	0-99
x_2	Gender	Female, Male
x_3	Injury Type	Blunt, Penetrating
x_4	Blood Pressure	0-300
x_5	Respiration Rate	0-99
x_6	GCS Eye	1-4
x_7	GCS Verbal	1-5
x_8	GCS Motor	1-6
x_9	Head Severity	0-6
x_{10}	Face Severity	0-6
x_{11}	Neck Severity	0-6
x_{12}	Thorax Severity	0-6
x_{13}	Spine Severity	0-6
x_{14}	Abdomen Severity	0-6
x_{15}	Upper Extremity Severity	0-6
x_{16}	Lower Extremity Severity	0-6
x_{17}	External Severity	0-6
y	Discharge Status	Dead, Alive

We use the data of NTDB for extracting such features. For instance, the feature x_1 corresponds to the age of a patient. Since patients having more than 89 years in the NTDB were classified in one class, we assign to these patients the age of 95 as default. The Injury Type (feature x_3) distinguishes between blunt trauma (i.e., physical trauma such as bone fracture) and penetrating trauma (i.e., when an object pierces the skin). The Blood Pressure (feature x_4) is given in mmHg and ranges from 0 to 300, while the Respiratory Rate (feature x_5) is measured in breaths per minute. Features x_6, x_7, x_8 refer to the Glasgow Coma Scale (GCS) [4] and examine the eye, the verbal response, and the motor skill of a patient according to the conditions listed in Table 2.

Features from x_9 to x_{17} evaluate the injury severity in different body regions according to AIS scale: 0 score means no injury, 1 a minor injury, 2 a moderate one, 3 a serious one, 4 a severe one, 5 a critical one, and 6 an unsurvivable injury.

To be able to compare the results with the TRISS method we excluded patients having no ISS or an ISS equal to zero. Since the NTDB distinguishes nine body regions while TRISS considers only six regions we applied the rules defined in [35] to assign the AIS scores to the six body regions. Moreover, we discarded the patients having a non-consistent ISS (i.e., the patients for which the computed ISS score differs from that of NTDB). Each patient is labelled with a Discharge Status indicating whether the patient survived or not.

We finally ended up with a dataset Δ of 656,092 records, each of which characterised by 17 different features (15 numerical and 2 nominal). Furthermore, for testing the prediction model scalability and evolution when increasing the dataset size, we considered also six different subsets $\Sigma_1 \subset \cdots \subset \Sigma_6 \subset \Delta$. For $k = 1, \ldots, 6$ each Σ_k contains $100000 \times k$ instances. In particular for $k = 2, \ldots, 6$ every Σ_k was created by adding to the previous Σ_{k-1} dataset 100,000 different instances randomly chosen from Δ. The datasets composition is summarized in Table 3.

Table 3: Datasets composition.

Dataset	No. of Patients	Alive Patients	Dead Patients
Σ_1	100000	95304	4696
Σ_2	200000	191455	8545
Σ_3	300000	287434	12566
Σ_4	400000	383237	16763
Σ_5	500000	478603	21397
Σ_6	600000	574355	25645
Δ	656092	627099	28993

3.2 Validation

In order to validate and test each classifier we used a 10-fold cross validation [36], a standard practice in the ML community for avoiding overfitting problems. Each dataset $D \in \{\Delta, \Sigma_1, \ldots, \Sigma_6\}$ we considered has been randomly partitioned in 10 disjoint subsets $D_1, \ldots, D_{10}$ called folds. Then, for $i = 1, \ldots, 10$ we used a fold D_i as the test set and the union of the remaining folds $\bigcup_{j \neq i} D_j$ as the training set.

Following the standard terminology [37] four possible outcomes are possible when evaluating a binary classifier that classifies a given instance as "positive" or "negative":

- *true positive* or hit, when the instance is positive and it is classified as positive;
- *true negative* or correct rejection, when the instance is negative and it is classified as negative;
- *false positive* or false alarm, when the instance is negative and it is classified as positive;

Table 2: Test scores for the GCS (Glascow Coma Scale)

Score	Eye	Verbal	Motor
1	No eye opening	Makes no sounds	No motor response
2	Eye opening to pain	Incomprehensible sounds	Extension to pain
3	Eye opening to voice	Utters inappropriate words	Flexion to pain
4	Eyes open spontaneously	Confused, disoriented	Withdrawal from pain
5	N/A	Oriented, converses normally	Localising pain
6	N/A	N/A	Obeys command

- *false negative* or miss, when the instance is positive and it is classified as negative;

In our case, a positive instance is a surviving patient while a negative one is a non-surviving patient.

In order to evaluate the performance of the ML classifiers we considered some standard evaluation metrics. Formally, if T_P, T_N, F_P, and F_N are respectively the number of true positive, true negative, false positive, and false negative cases identified by the classifier over the test set, we measure the classification effectiveness in terms of:

- *Accuracy (ACC)*: considers the correctly classified cases by taking into account both survivors and non-survivors. Formally,

$$ACC = \frac{T_P + T_N}{T_P + F_P + F_N + T_N}$$

- *Sensitivity* or *True Positive Rate* (TPR): is the ratio between patients classified as survivors and people that actually survived. This measure, also known as *hit rate* or *recall*, is given by:

$$TPR = \frac{T_P}{T_P + F_N}$$

- *Specificity* or *True Negative Rate* (TNR): is the ration between patients classified as non-survivors and people that actually did not survive:

$$TNR = \frac{T_N}{F_P + T_N}$$

- *Kappa* (κ): measures inter-rate agreement between classifiers, taking into account also the agreement occurring by chance [38]. It is a value between -1 and 1, where a bigger κ means better accuracy and reliability. Formally,

$$\kappa = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)}$$

where, denoting with n the number of test instances, we have that:

$$\Pr(a) = \frac{T_P + T_N}{n}$$

is the relative observed agreement;

$$\Pr(e) = \frac{(T_P + F_N) \cdot (T_P + F_P)}{n^2} + \frac{(F_N + T_N) \cdot (F_P + T_N)}{n^2}$$

is the hypothetical probability of chance agreement.

For the classification models that naturally lead to probabilistic esteems of class membership (i.e., all the previously described methods except SVMs), we used the default threshold of 0.5 to predict if an instance belongs to the positive or negative class. To better evaluate what happens when different thresholds are considered, we also plotted the *Receiver Operating Characteristic* (ROC) curves. ROC is a plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the sensitivity against the false positive rate at various threshold settings. The false-positive rate, also known as the fall-out, can be calculated as 1 - specificity. Starting from the ROC curve we computed also the *Area Under the Curve* (AUC) that, as the names implies, is the area under the ROC curve. This value represents the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative example.

3.3 System and Tools

The experiments were conducted by using Intel Dual-Core 2.93 GHz computers with 3 MB of CPU cache, 2 GB of RAM, and Debian 3.2 operating system. Computation times were tracked thanks to the Unix `time` command. We evaluated several off-the-shelf classifiers by means of WEKA [39], a software including many data mining algorithms for classification, clustering, etc. In particular, we tested a number of WEKA classifiers implementing the algorithms described in Section 2.3, namely:

- *KNN* (`weka.classifiers.lazy.IBk`), a k-Nearest Neighbours classifier;
- *J48* (`weka.classifiers.trees.J48`), (un)pruned C4.5 decision trees;

- *Logistic* (`weka.classifiers.functions.Logistic`), multinomial logistic regression model with a ridge estimator
- *Naive Bayes* (`weka.classifiers.bayes.NaiveBayes`), Naive Bayes classifier using estimator classes;
- *Neural Net* (`weka.classifiers.functions.MultilayerPerceptron`), uses Artificial Neural Networks to classify instances;
- *PART* (`weka.classifiers.rules.PART`), exploits PART decision lists;
- *Random Forest* (`weka.classifiers.trees.RandomForest`), constructs a random forest of decision trees;
- *SMO* (`weka.classifiers.functions.SMO`), implements SMO algorithm for training a support vector classifier.

Note that all these algorithms are highly parametrisable, i.e., they can be configured in several different ways by tuning the algorithm parameters. For example, it is possible to set the number of trees for Random Forest, the complexity constant C for SMO, or the neighbourhood size k of k-NN. Testing all the possible parameters configurations is outside the scope of this paper. To avoid biases [40], we decided to test all the algorithms with their WEKA default parameters.[3]

However, as better detailed in Section 4.11, in this paper we also provide some examples of parametrization.

4 Results

According to the methodology described in Section 3 we compared the performance of TRISS against the eight ML approaches listed in Section 3.3 by using the datasets introduced in Section 3.1 and the evaluation metrics of Section 3.2. Tables 4, 5, 6, and 7, show respectively the classification accuracy, the sensitivity, the specificity, and the κ statistic for each method and dataset. Every table is sorted from top to bottom according to the method performance on the whole dataset Δ.

4.1 Accuracy

Looking at Table 4, the approach having greater classification accuracy on Δ is Random Forest (97.74%), followed by J48 (97.26%) and Neural Network (97.18%). Conversely, TRISS is second-last with an accuracy of 96.47%, about 1.27% less than Random Forest (i.e., about 8332 patients less). Figure 2 shows more clearly the behaviours of the tested methods as the dataset size increases. Only Random Forest (shortly, RF) and KNN approach have a monotonic performance, and in particular RF is the best approach for each considered dataset. The accuracy of J48, NeuralNet, PART, SMO, and Logistic are quite similar. In particular, it is interesting to see that the SMO performance is pretty much the same of the Logistic approach. Figure 2 clearly depicts the accuracy difference between RF and TRISS, and in particular the performance divergence as the number of patients increases.

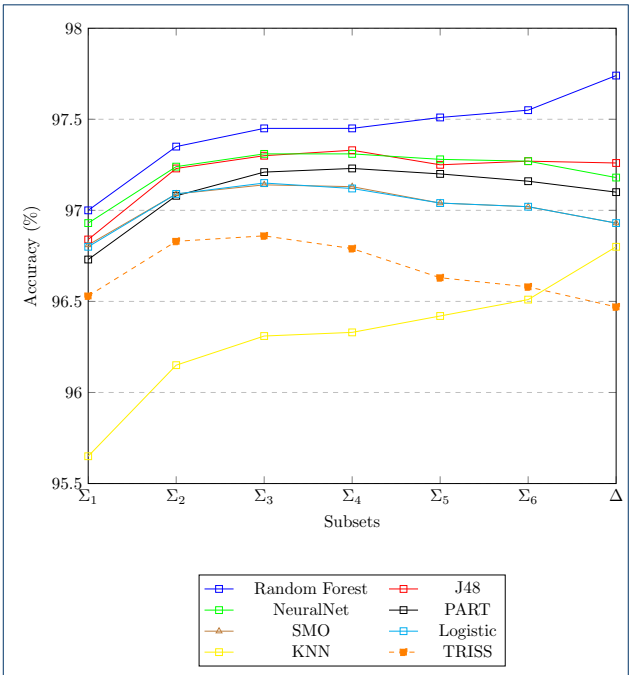


Figure 2: Accuracy results. For readability, Naive Bayes approach is not included.

4.2 Sensitivity

Table 5 shows the sensitivity results, i.e., the ratio between patients classified as survivors and people that actually survived. With this measure SMO is the overall best approach, i.e., the best classifier in predicting the alive patients. It outperforms all the other methods, with an almost constant performance ranging between 99.67% and 99.71%. RF, NeuralNet, Logistic, J48, and PART have a similar performance while KNN, TRISS, and Naive Bayes have a lower sensitivity. TRISS sensitivity tends to decrease as the number of patients increases. In particular, the performance difference on Δ between SMO and TRISS is evident: 1.56%, i.e., about 10235 patients.

4.3 Specificity

Specificity results are reported in Table 6. As can be seen, results are in sharp contrast w.r.t. those previously reported. Naive Bayes —clearly the worst approach

[3] We used the 3.7.12 version of WEKA.

Method	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Δ
Random Forest	97	97.35	97.45	97.45	97.51	97.55	**97.74**
J48	96.84	97.23	97.3	97.33	97.25	97.27	97.26
NeuralNet	96.93	97.24	97.31	97.31	97.28	97.27	97.18
PART	96.73	97.08	97.21	97.23	97.2	97.16	97.1
SMO	96.81	97.09	97.14	97.13	97.04	97.02	96.93
Logistic	96.8	97.09	97.15	97.12	97.04	97.02	96.93
KNN	95.65	96.15	96.31	96.33	96.42	96.51	96.8
TRISS	*96.53*	*96.83*	*96.86*	*96.79*	*96.63*	*96.58*	*96.47*
Naive Bayes	94.06	94.44	94.51	94.43	94.19	94.14	93.97

Table 4: Accuracy (%).

Method	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Δ
SMO	99.67	99.69	99.7	**99.71**	99.7	99.69	99.68
Random Forest	99.34	99.39	99.41	99.41	99.41	99.4	99.44
NeuralNet	99.23	99.28	99.31	99.36	99.4	99.29	99.36
Logistic	99.22	99.29	99.31	99.3	99.28	99.27	99.25
J48	99.16	99.24	99.26	99.29	99.25	99.26	99.22
PART	99.12	99.21	99.3	99.32	99.37	99.26	99.19
KNN	97.96	98.2	98.28	98.29	98.31	98.36	98.48
TRISS	*98.44*	*98.52*	*98.51*	*98.42*	*98.27*	*98.21*	*98.12*
Naive Bayes	95.21	95.5	95.58	95.48	95.24	95.18	95.03

Table 5: Sensitivity (%).

according to accuracy and sensitivity— is now by far the best and more stable approach. Its performance ranges between 70.11% and 71.01%, and it significantly outperforms the TRISS performance on the whole dataset Δ (10.29%, about 67511 patients). Conversely, the approach with best sensitivity (SMO) is now the approach with worst specificity. In other terms, there is a sort of symmetry: SMO is the best approach in predicting alive patients and the worst in predicting dead patients while Naive Bayes works in a diametrically opposite way. In contrast, RF shows a good robustness and quality of performance on the whole dataset Δ, where it is also better than TRISS. In particular, the performance of RF is monotonically increasing from 49.62% of Σ_1 to 61.01% of Δ.

We would like to remark that from our perspective it is probably more ethical to avoid false negatives since we do not want to classify patients that can live as dead, and therefore do not attempt to save them. On the other hand, it is also important to not be too cautious: classifying a dead patient as alive could result in a waste of time and resources, perhaps at the expense of other critical —but still savable— victims.

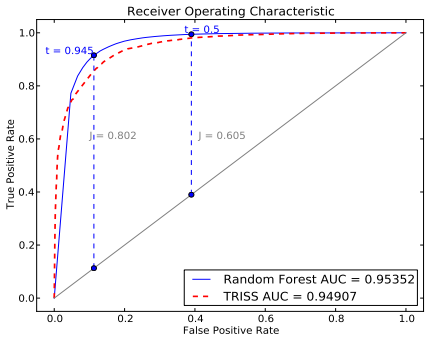
4.4 Kappa statistics

The Kappa statistics reported in Table 7 show that all the approaches have positive κ values, in particular the tree-based approaches (RF and J48). This is rather important because having a predictor with a κ greater than 0 means that it is possible to be better than making the prediction randomly. The results of the κ statistic are somehow similar to the accuracy results with few swaps in the rankings: RF and J48 are still the best two methods, while Naive Bayes is the worst. Please note that the κ statistic is generally thought to be a more robust measure than simple percent agreement calculation since it takes into account the agreement occurring by chance. This is important for skewed datasets like ours, in which the number of survivors is far greater than the number of dead patients.

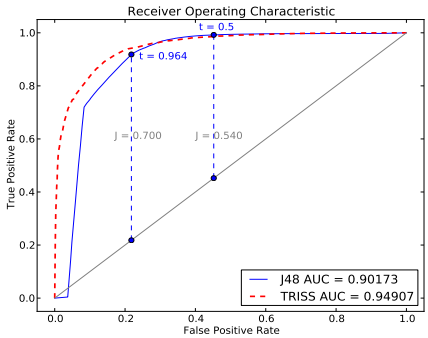
4.5 ROC and AUC

As previously stated, all the previous results have been obtained with the default parameters of WEKA. As for the TRISS method, the default threshold used to discriminate if an instance belongs to the positive or negative class is set to 0.5. Clearly, different results can be obtained when different thresholds are used. To show how the different ML approaches behave when the threshold changes, we computed their ROC curves and their AUCs. To do so, with the only exception of SMO, we used the esteems of the class membership probabilities given by the classifiers. For the SMO classifier only, we employed the Platt scaling method [41], i.e, a well-known technique that associates to every prediction its class membership probabilities by fitting a logistic regression model.[4] This was necessary because SMO alone does not provide class membership probabilities.

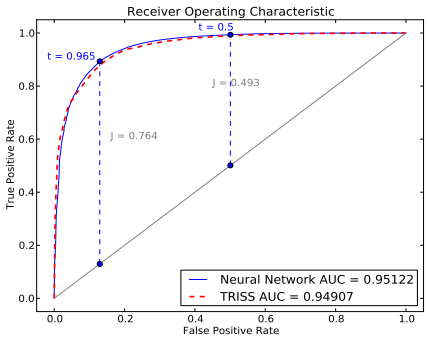
[4] In WEKA this can be done with the "-M" option of the SMO classifier.



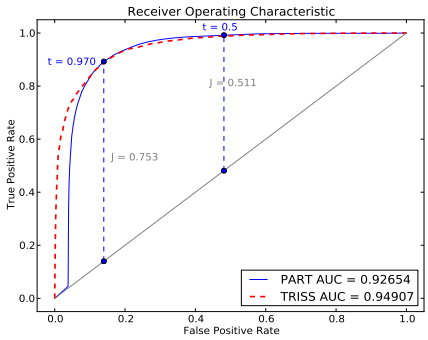
(a) Random Forest



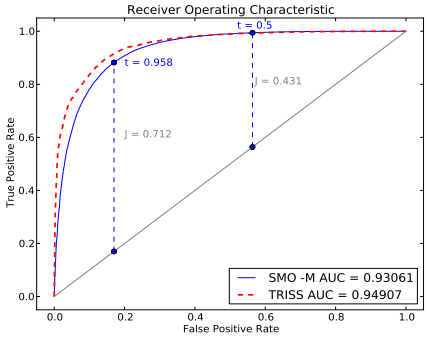
(b) J48



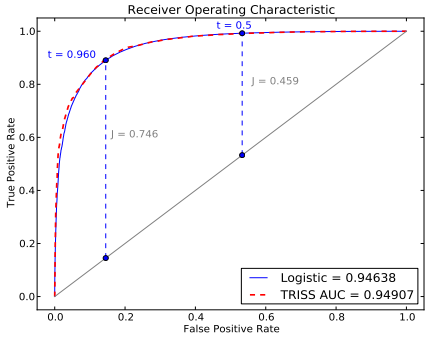
(c) Neural Net



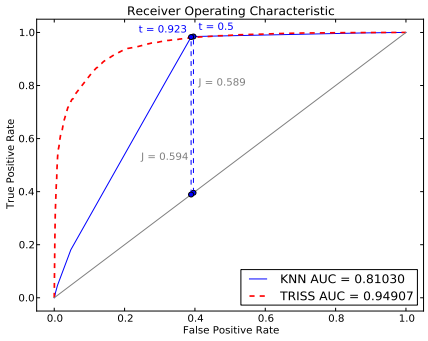
(d) PART



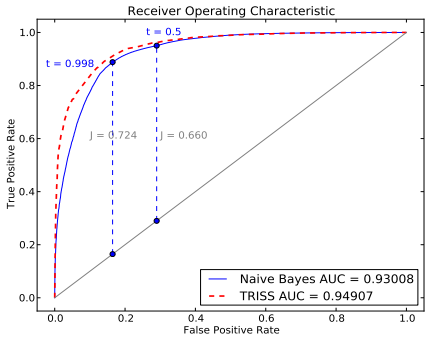
(e) SMO



(f) Logistic



(g) KNN



(h) NB

Figure 3: ROC curves

Method	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Δ
Naive Bayes	70.7	70.78	70.11	70.42	70.72	70.84	**71.01**
Random Forest	49.62	51.49	52.5	52.66	54.84	56.17	61.01
TRISS	*57.88*	*58.89*	*59.12*	*59.61*	*59.99*	*60.16*	*60.72*
KNN	48.66	50.26	51.38	51.63	54.16	55.06	60.42
J48	49,72	52,14	52,44	52,31	52,35	52,69	54.79
PART	48,17	49,35	49,4	49,51	48,77	50.16	51,89
NeuralNet	50.17	51,53	51,5	50.43	49,86	52,05	49,92
Logistic	47.76	47.71	47.55	47.25	46.96	46.54	46.7
SMO	38,65	38,76	38,6	38,05	37.72	37.3	37.58

Table 6: Specificity (%).

Method	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Δ
Random Forest	0.5937	0.6107	0.6199	0.621	0.6407	0.6503	0.6931
J48	0.5808	0.6025	0.6061	0.6077	0.6055	0.6086	0.6246
KNN	0.4896	0.5072	0.5193	0.5221	0.5453	0.556	0.6084
PART	0.564	0.5762	0.5834	0.5861	0.5851	0.5876	0.5976
NeuralNet	0.5898	0.6008	0.6026	0.5981	0.597	0.6065	0.5958
TRISS	*0.5924*	*0.5970*	*0.5959*	*0.5921*	*0.5860*	*0.5830*	*0.5845*
Logistic	0.5681	0.5689	0.5686	0.5648	0.5614	0.5572	0.558
SMO	0.518	0.5196	0.5182	0.5091	0.5133	0.5042	0.5065
Naive Bayes	0.4982	0.4937	0.4901	0.487	0.4818	0.4796	0.4805

Table 7: Kappa statistics.

Figure 3 reports the ROC curves obtained by applying a 10-fold cross-validation on the entire dataset Δ. Each classifier is compared with the ROC curve of the TRISS method. In every plot we marked the point obtained with threshold $t = 0.5$ and the point that maximizes the *Youden's J* statistic [42], i.e., the point where $J = sensitivity + specificity - 1$ is maximal.

Considering the AUC it is immediately visible that the best method is Random Forest having an AUC of 0.953. TRISS is better than Random Forest for thresholds close to 1, while the difference between the two approaches is negligible for thresholds lower than 0.4. These thresholds, however, denote settings that are almost useless in a real scenario since they represent the extreme cases where we assign almost every instance to either the positive or negative class. Random Forest instead is able to overcome TRISS when the specificity and the sensitivity are close to 1. The maximal J reached by Random forest is 0.8025 with a threshold of 0.9449, a sensitivity of 0.9152, and a specificity of 0.8873. For TRISS, instead, the maximal J is 0.750375 with a threshold of 0.954555, sensitivity 0.882268, and specificity 0.868106.

Usually, the point having maximal J is considered a good candidate for deciding which threshold to use. However, since there are other Pareto-optimal points, other possible thresholds might be adopted. In our case, since we deem sensitivity more important than specificity, we believe that using as threshold the Youden's J criteria is not a particularly good choice. In our opinion, it is better to use a higher threshold (e.g., 0.5 or a bit more) to obtain a higher sensitivity and a still reasonable specificity.

Looking at the other plots, we can see that only Random Forest and NeuralNet have an AUC bigger than TRISS. This is due to the fact that, compared to the TRISS approach, the classifiers usually have a lower sensitivity for threshold very close to 1. But when the threshold decreases all the approaches except Naive Bayes are able to (sometime only slightly) overcome the TRISS approach.

All the ROC curves have a similar shape, with the only exception of KNN. This method has indeed fewer points in the leftmost part of the plot. We believe that this is due to how KNN computes the probability distribution. Indeed, each k-nearest neighbor first votes for its respective class label. These votes are then averaged and normalized to create a probability distribution. In our setting, since the majority class is overrepresented, it is therefore unlikely to find in the neighborhood a high number of instances voting for the minority class.

4.6 Calibration Curve

While receiver operating Characterictic (ROC) describes the ability of the model to distinguish between patients who died and patients who survived, Calibration curves on the other hand show how much a model is reliable by measuring how close the estimates are to

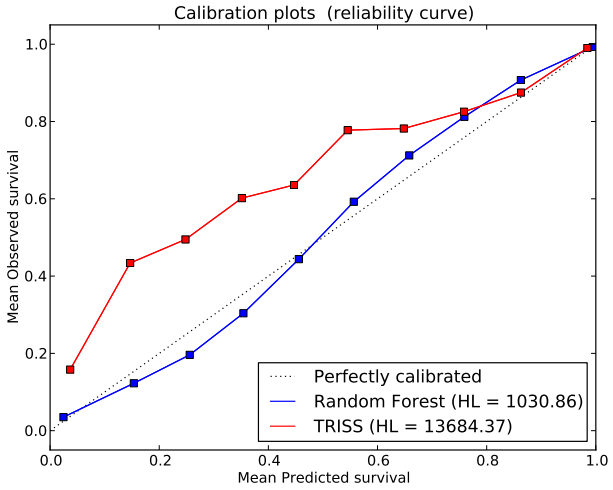


Figure 4: Calibration Curve (Random Forest vs TRISS).

the real probabilities. Calibration curves are constructed by plotting the observed binary events against their predicted probability. An optimally calibrated model follows the dotted diagonal line. The closer the curve is to the diagonal line, the better the model.

First, the observations are sorted in an increasing order of their predicted probability. Then, the observations are divided into 10 groups (subdivision based on 10 fixed thresholds $(0.1, 0.2, 0.3, \cdots, 1.0)$). For each group, we compute the mean observed and the mean predicted survival.

In order to evaluate the goodness-of-fit of the prediction models, we used the Hosmer-Lemeshow (HL) statistic [43]. The smaller the value of HL statistic, the better is the calibration.

The Hosmer-Lemeshow test statistic is given by [44]:

$$HL = \sum_{j=1}^{10} \frac{(O_j - E_j)^2}{E_j(1 - E_j/n_j)}$$

where:
n_j = Number of observations in the j^{th} group.
$O_j = \sum_i y_{ij}$ = Observed number of cases in the j^{th} group.
$E_j = \sum_i p_{ij}$ = Expected number of cases in the j^{th} group.

Figure 4 shows the calibration curves for the TRISS method and the best classifier (i.e., Random Forest). One can observe here that RF is well calibrated. We can see that the RF predictions are much closer to the observed survival than the TRISS predictions. For the TRISS method, the observed probabilities are higher than the predicted values, and the difference is largest for patients with a predicted survival between 0.1 and 0.7. Moreover, the value of the HL statistic of RF (reported in the legend) was significantly better than the one of the TRISS model (the smaller the better).

4.7 Scalability

In Table 8 we report the time needed for performing the whole cross-validation. With the only exception of the KNN method —that does not build a prediction model— the computational times refer basically to the time needed for building the prediction model. Once the model is built, the class prediction for a new instance is instantaneous. Conversely, for the KNN approach the computational time refers to the prediction time since no training is required and no prediction model is learned. In any case, for all the considered approaches, the class prediction for a given instance is always computed in few milliseconds.

Naive Bayes and Logistic are the fastest approaches while PART and SMO require a long training phase (more than half a day in some cases). We remark that this is however just the training time, which is performed off-line once in a while. For this reason we think that all the studied methods can be used in real life scenarios where the prediction models can be updated periodically (e.g., once in a month) to learn from new incoming patients. All the approaches are scalable enough to be used and trained with hundred of thousand of patients.

In summary, we can say that for all the performance measures and all the datasets we considered there is always a ML method able to outperform the TRISS methodology. Looking at the whole dataset Δ, the most accurate classifier is Random Forest. SMO is the best approach in predicting alive patients, while Naive Bayes outperforms all the other approaches if we consider the prediction of dead patients. SMO uses a cautious approach: it performs poorly according to accuracy and specificity, but has a high sensitivity. This is reasonable, since classifying a dead patient as alive might not have the same bad impact of classifying an alive patient as dead.

Overall, we can say that the best approach we evaluated is Random Forest: it has the best accuracy, AUC, and κ-statistic, as well as the second-best sensitivity and specificity. Furthermore, its performance monotonically increases as the dataset size grows, meaning that it can be very effective to exploit incoming knowledge. Moreover, considering the whole dataset Δ, it is always better than TRISS. Finally, its training time is not negligible but however fairly reasonable.

Method	Σ_1	Σ_2	Σ_3	Σ_4	Σ_5	Σ_6	Δ
Naive Bayes	12s	23s	33s	45s	57s	1min13s	1min18s
Logistic	55s	1min19s	2min4s	3min3s	5min37s	7min1s	7min46s
J48	7min43s	22min2s	41min48s	32min7s	48min24s	63min56s	74min43s
NeuralNet	38min28s	77min1s	73min57s	96min12s	119min43s	146min9s	157min53s
Random Forest	31min13s	42min50s	74min1s	108min6s	151min58s	186min2s	360min9s
PART	24min20s	42min49s	100min4s	182min47s	710min6s	1057min26s	818min6s
KNN	38min13s	154min37s	206min9s	351min19s	956min30s	1367min44s	1549min36s
SMO	79min4s	131min37s	310min49s	575min12s	963min25s	1429min3s	1811min44s

Table 8: Computational times.

4.8 Imbalanced data

Our dataset is highly unbalanced: slightly less than 5% of the people is classified as a dead patient. Accuracy in this case may not be the best metric to use due to the accuracy paradox [45] stating that models with low accuracy may have greater predictive power than models with higher accuracy. For example, considering our dataset, a classifier that always assigns a patient to the alive class has an accuracy of more than 95%.

There are several options for learning from unbalanced data. The most common solutions are to discard some training instances of the most represented class (*undersampling*) or, dually, to replicate some instances of the less represented class (*oversampling*).

Due to the large size of our dataset we performed the undersampling. We used the `FilteredClassifier` implementation of WEKA and we applied a 10-fold cross-validation in which for every training set we randomly discarded some positive instances.

Table 9 shows the accuracy, sensitivity, specificity, and kappa statistics of the 10-fold cross validation applied on the Δ dataset. For every metric we report its value and its difference w.r.t. the value obtained without undersampling (cf. Tables 4-7).

It is immediately visible that when undersampling is used the accuracy, the sensitivity, and the kappa statistic decrease while the specificity increases. The differences can be very significant. If we exclude Naive Bayes where differences are not pronounced, for all the classifiers the accuracy decreases by more than 7% and the sensitivity by more than 9%. This means that when undersampling is used a lot of patients that could live will be predicted as people that are going to die. In our opinion, these errors may be too numerous to justify the use of the undersampling because, as previously stated, it is probably more ethical to avoid false negatives. On the other hand, undersampling increases the specificity and this could be helpful in disasters or emergencies situations where only a fraction of the patients that could live can be saved. In these cases predicting with higher accuracy which patients will die may become more critical than predicting the patients that will live.

4.9 Critically Injured Patients

Our dataset presents another unbalance related to the number of patients that are severely injured w.r.t. to the patients that have only moderate injures. In our benchmark Δ, 80% of the whole population has ISS ≤ 16, indicating that they are moderately injured [46]. Table 10 shows the ratios and mortalities in the 2 groups. As expected, we can see that the mortality is higher in the first group of severely injured patients (ISS > 16), and lower in the second group (ISS ≤ 16).

	Population	Mortality
ISS ≤ 16	80%	1.36%
ISS > 16	20%	16.66%
All	100%	4.4%

Table 10: Statistics for severely injured patients.

Since it may be more important to have reliable prediction on severely injured patients, the unbalance may constitute a thread to the validity of our results. The huge number of moderately ill patients may indeed worsen the prediction performances for other, more relevant, classes.

To show that this is not the case, we report on Table 11 the performance of Random Forest and TRISS when we are restricting only to the classes of severely injured patients. We can notice that the accuracy, sensitivity and specificity of RF is quite the same for patients having ISS > 16, which means that the approach is not very influenced by the minimally injured patients. The accuracy and sensitivity of Random Forest is higher than TRISS. The specificity of TRISS instead is slightly better than RF. This is probably due to the fact that TRISS in this case predicts a lot of patients as dead.

4.10 Clusterization of Old Patients

Another thread to the validity of our results is due to the fact that the NTDB database does not discriminate old patients having more than 89 years old since it clusters them in a unique group. Unfortunately, age is

Method	Accuracy		Sensitivity		Specificity		Kappa	
	Value	Diff.	Value	Diff.	Value	Diff.	Value	Diff.
Naive Bayes	92.75	-1.72	93.54	-2.02	75.71	4.70	0.4466	-0.0339
NeuralNet	89.84	-7.34	89.96	-9.40	87.29	37.37	0.3910	-0.2048
J48	88.83	-8.43	88.86	-10.36	88.15	33.36	0.3681	-0.2565
Random Forest	89.18	-8.56	89.11	-10.33	90.70	29.69	0.3839	-0.3092
SMO	89.86	-7.07	90.08	-9.60	85.21	47.63	0.3855	-0.1210
Logistic	89.48	-7.45	89.64	-9.61	85.91	39.21	0.3775	-0.1805
PART	88.39	-8.71	88.46	-10.73	86.73	34.84	0.3536	-0.2440
KNN	86.48	-10.32	86.51	-11.97	85.90	25.48	0.3115	-0.2969

Table 9: 10-fold cross-validation with undersampling on Δ.

	Accuracy		Sensitivity		Specificity	
	RF	TRISS	RF	TRISS	RF	TRISS
ISS > 16	**93.08%**	86.84%	**99.44%**	*90.4%*	61.22%	**69.08%**
All	**97.74%**	96.47%	**99.44%**	*98.12%*	**61.01%**	60.72%

Table 11: Performance for Random Forest and TRISS Models for severely injured patients.

an important predictor for the of outcome of a trauma. Mortality indeed increases between the ages of 45 and 55 for the same injury severity and is doubled above 75 years. As can be seen from Table 12, the mortality ratio for the patients having more than 89 years is very different from the one of younger patients and this difference may impact the prediction performances.

	Population	Mortality
Age ≤ 89	98%	4.33%
Age > 89	2%	8.64%
All	100%	4.4%

Table 12: Statistics for patients based on their age.

Luckily, as happens with the critically injured patients, RF has been proven to be quite robust. Table 13 shows the performance of Random Forest and TRISS when we restrict ourself to consider only the patients for which we know their true age (i.e., patients having less than 90 years). We can clearly notice that the performance of RF and TRISS stay almost the same. The only difference is that the specificity of the TRISS method decreases when the eldery patients are not considered and is surpassed by the specificity of Random Forest. This is expected since the TRISS predicts usually more people as dead and the mortality rate for elderly patients is higher than the mortality rate for younger patients.

4.11 Parameter Tuning

Machine learning classifiers usually comes with a lot of parameters that can be set. Trying all the possible values is almost impossible due to the combinatorial explosion of the parameter settings. Moreover, the tuning of the parameters can lead to an overfitting of the model and to a selection bias [40]. For these reasons, the results presented so far are obtained by using the WEKA default parameters of each classifier.

However, in this section we would like to provide an example of parameter tuning. We take the best "default" classifier (i.e., Random Forest) and we show how it can be improved by tuning its parameters. In particular, we varied the number of trees employed by Random Forest.

Table 14 reports the results of the 10-fold cross validation when Random Forest with 32, 64, 128, 256, 512, and 1024 trees are used. Since the differences in percentage are small, for presentation purposes we also show the absolute difference between the tuned approach and the default one.

The performance tends to increase as the number of trees increases. However, after using 256 trees we see that the accuracy still improves but its increase is due to the increment of the specificity while the sensitivity decreases. As expected, since Random Forest is an ensemble method that averages over many trees, after a certain point even the accuracy stabilizes and the cost of collecting a large sample of trees becomes higher than the benefit in accuracy obtained from such larger sample of trees.

Overall, the increments in performance are modest (please note that 50 patients are less than 0.008% of the population).

Clearly, it is difficult to decide a priori what is the best number of trees to use since the use of a large number of trees may cause overfitting and longer training times. In general, the problem of selecting the best configuration of parameters is a challenging task which is outside the scope of this paper.

	Accuracy		Sensitivity		Specificity	
	RF	TRISS	RF	TRISS	RF	TRISS
Age $\leq$ 89	**97.77%**	96.55%	**99.44%**	98.1%	61.02%	62.36%
All	**97.74%**	96.47%	**99.44%**	*98.12%*	**61.01%**	60.72%

Table 13: Performance for Random Forest and TRISS Models for patients with less than 90 years.

Trees	Accuracy		Sensitivity		Specificity		Kappa
	Value	Patients	Value	Patients	Value	Patients	Value
32	97.6965	-279	99.4310	-37	60.1800	-242	0.6861
64	97.7285	-69	99.4367	-1	60.7801	-68	0.6913
100	97.739	0	99.4369	0	61.0147	0	0.6931
128	97.7444	35	99.4399	19	61.0699	16	0.6938
256	97.7456	43	99.4402	21	61.0906	22	0.694
512	97.7467	50	99.4390	13	61.1424	37	0.6943
1024	97.7428	25	99.4357	-8	61.1289	33	0.6939

Table 14: Results of 10-fold cross-validation by varying number of trees for Random Forest on Δ. The default number of trees is 100.

5 Trauma Survival Prediction Tool

On the top of the best classifiers evaluated in Section 4 we developed a C++ tool for predicting if a patient will survive or not according to his/her features. This tool can help practitioners to predict the survival probability for a given trauma patient. It might be used for replacing, or maybe complementing, the TRISS methodology.

Figure 5 shows a screenshot of the calculator interface. The graphical interface is rather simple and allows the user to define the value of the features listed in Table 1. To guide the user, every field has restricted values that are presented for the user convenience. For instance, the field abdomen (first on the right column), presents the user with 6 different choices linking the numerical number with a short description of the meaning of the numerical value. As example, for the value 4 selected the user is reminded that this corresponds to a severe abdomen injury.

When all the fields are set, the user can choose one of the available classifiers (viz. Random Forest, NeuralNet, J48, SMO, Naive Bayes) to compute the prediction if the patient will live of die. The choice of the prediction algorithm to use is done by using a selection button. The algorithm is then executed when the user press the "Predict" button. The tool in few milliseconds returns the predicted class (i.e., dead or alive). Moreover, it also gives an estimation of the probability of the outcome based on the selected machine learning approach.

The tool comes with a prediction model for all the machine algorithms supported. The models are generated off-line by using the entire dataset Δ obtained from the NTDB database and better described in Section 3.1. The algorithm and implementation used are exactly those presented and used in the previous section. For this reason, if the patients to examine present the same statistical patterns of those recorded in the NTDB database, the accuracy of the predictions should follow the one presented in the previous section. Thanks to the results obtained by using the cross-validation as validation mechanism, we are fairly confident that the models are general enough and not overfitting the data.

The tool is publicly available at https://github.com/imanesefrioui/tool_prediction_survival_executable. Since the tool relies on WEKA classifiers and Qt libraries, it requires the Java Runtime Environment [47], the installation of WEKA and the Qt libraries. All these dependencies are available for free.

6 Related work

Several approaches relying on Machine Learning and Artificial Intelligence techniques have been used in medicine to predict diseases and survival probabilities.

Schetinin et al. [15] proposed a solution based on Bayesian Model Averaging over Decision Trees to predict survival of trauma patients. Their technique consists of drawing n samples of decision trees using a Monte Carlo Markov Chain method. They predictive survival probability is then computed by using the n decision trees generated and the Bayes rule. However, as the authors said, the Bayesian risk assessments are computationally expensive and further research should be conducted to make this method tractable for large scale datasets.

In [48] a Neural Network model is used to predict the probability of survival of trauma patients. They used the same features listed in Table 1. Similarly to what we did, they proposed a Multilayer Perceptron model with three layers, one single hidden layer

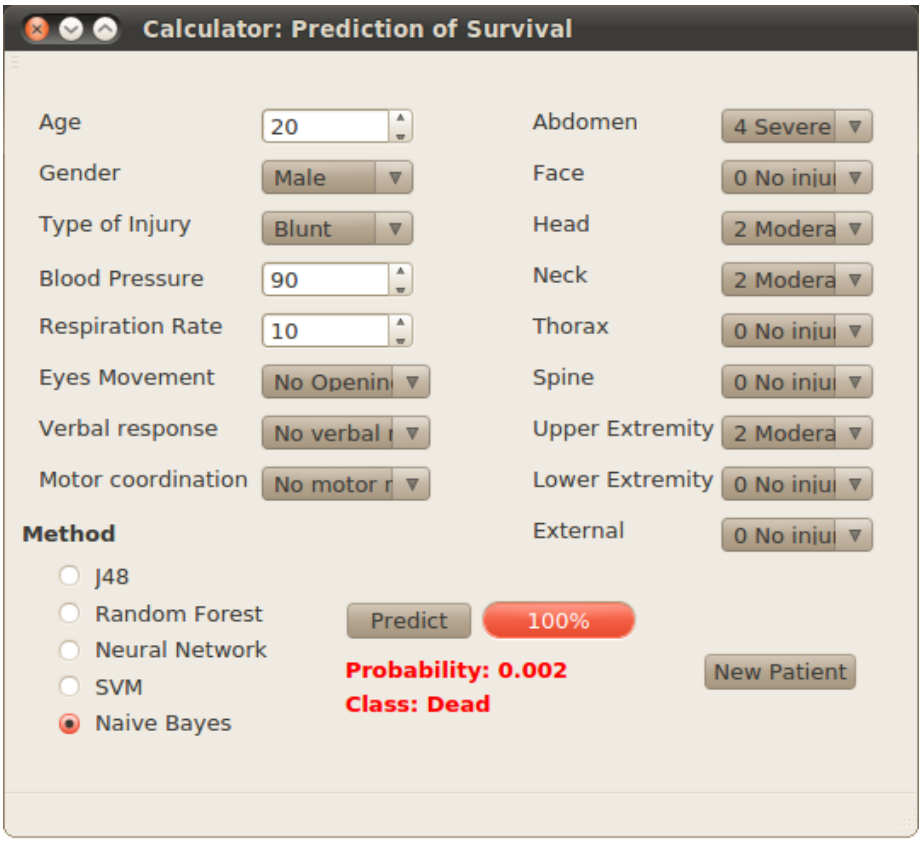


Figure 5: Screenshot of the tool for survival prediction.

and two hidden units by using Back Propagation and Quasi-Newton method for training their model. They show that neural networks yield better results than the TRISS method, but, unlike us, they did not compare their method with other promising techniques such as SVMs.

Similarly, in [49] authors use a Neural Network model to predict the mortality in intensive care units. Their approach was compared with a logistic regression model on a total of about 13000 adult patients that were randomly divided into training (66%) and test (33%) sets. The two methods were evaluated in terms of Receiver Operator Characteristic (ROC) curves, and experimental results have shown that Neural Network outperformed the logistic regression.

A Multilayer Perceptron is also used in [50] to predict heart disease using factors such as family history, smoking, blood pressure. In [51] authors propose a method for classifying medical brain images as normal or abnormal according to a Neural Network scatter search. These approaches have however a different goal since we are not concerned with predicting risk diseases.

Sujin et al. [52] compared the intensive care unit mortality prediction built with different Machine Learning techniques on a database of the University of Kentucky Hospital. They used about 38000 patients, 15 features, and different ML approaches, namely: Artificial Neural Networks, Support Vector Machines, Decision Trees and Logistic Regression. These methods have been implemented in a software tool [53]. In their experiments the best performing model was the Decision Tree model, followed by SVM, Logistic Regression, and ANN. The motivations of their study are similar to ours, even though we had to face different parameters and a large dataset of patients.

In [54], Random Forest was used for classify trauma patients into two categories: upper level admission (i.e., patients requiring level I trauma center care) and lower level admission (i.e., patients not requiring level I trauma center care). However, differently from our study, they used 83 attributes and less than 2000 patients. Moreover, they did not compare their findings w.r.t. other classifiers.

In [55] Naive Bayes, DT, SVM and ANN were used to predict the survival of burn patients. In their case the best predictor was the Naive Bayes approach. Even

though they rely like us on the Weka framework for building the predictors, they limited their study only to burn patients using only 180 patients with 10 features (viz., age, gender, and percentages of burn in the eight body regions).

As far as the prediction methods are concerned we would like to recall also the Revised Injury Severity Classification (RISC) introduced in 2009 and later revised as RISC II [56] in 2014. RISC is a new prediction algorithm based upon 2008 severely injured patients from TR-DGU, the trauma registry of the German Society for Trauma. Contrary to TRISS, this score includes laboratory parameters such as base deficit, haemoglobin, and partial thromboplastin time, as well as interventions like cardiopulmonary resuscitation (CPR) [57]. It is based on a logistic equation that calculates the probability of survival by using the coefficients for 11 variables. Authors reports that the updated RISC II prognostic has outperform the RISC model and the TRISS in terms of discrimination, precision and calibration. Unfortunately, by relying on the data of the NTDB database only, the comparison between Random Forest and RISC II can not be performed since the NTDB has no attribute representing the hemoglobin or the partial thromboplastin time which are essential to compute the RISC. For this reason, finding a big database for evaluating, if statistically Random Forest are comparable with the RISC II, is left as a future work.

7 Conclusions

The surge of Machine Learning techniques has found fertile ground in medical sciences. In this paper we conducted a rigorous study applying the most common and used classification methods for predicting the survival of trauma patients. We used a set of simple and easy to get features, and a huge dataset of patients extrapolated from NTDB, the largest aggregation of US trauma registry data ever assembled.

Overall, we can say that the best prediction method tested is the Random Forest classifier. In a not negligible number of cases (i.e., thousands of patients) it is able to outperform the TRISS methodology, which is the standard technique currently used to predict the survival probability of trauma patients.

On the basis of these findings we developed a tool for computing the survival prediction by means of Random Forest and some other promising classifiers. We hope that this tool, maybe used in combination with other approaches, can help the practitioners to improve the accuracy of the survival predictions.

As future works we are planning to integrate these experiments by evaluating more Machine Learning approaches and new, different features. Interesting directions concern also the parameters configuration of the classifiers and the selection of the most informative features. It would be interesting also to integrate the tool with new functionalities, for example by embedding it in decision systems for disaster-management [58] and check what is the impact on the prediction accuracy when unreliable data gather quickly during the emergency is used. Another direction worth investigation is to use merge existing databases to make a comparison between off-the-shelf machine algorithms and other survival prediction tools such as the RISC II.

Acknowledgements

This research work was carried out with the support of the Erasmus Mundus program of the European Union. We would also like to thank the Committee on Trauma, American College of Surgeons for providing us the National Trauma Data Bank used to conduct the experiments.

Compliance with Ethical Requirements

Conflict of Interest

Imane Sefrioui, Roberto Amadini, Jacopo Mauro, Abdellah El Fallahi, and Maurizio Gabbrielli declare that they have no conflict of interest.

Author details

[1] Faculty of Sciences of Tetouan, University Abdelmalek Essaadi. [2] Department of Computing and Information Systems, The University of Melbourne, Australia. [3] Faculty of Informatics, University of Oslo, Norway. [4] National School of Applied Sciences of Tetouan, University Abdelmalek Essaadi. [5] Department of Computer Science, University of Bologna / Lab. Focus INRIA, Italy.

References

1. Gebhart ME, Pence R. START Triage: Does It Work? Disaster Management & Response. 2007;5(3):68 – 73.
2. Koenig KL, Schultz CH. Koenig and Schultz's Disaster Medicine: Comprehensive Principles and Practices. Cambridge medicine. Cambridge University Press; 2009. Available from: http://books.google.co.ma/books?id=IfpOtV-FAl4C.
3. Rehn M, Andersen JE, Vigerust T, Krüger AJ, Lossius HM. A concept for major incident triage: full-scaled simulation feasibility study. BMC Emergency Medicine. 2010;10(1):1–7. Available from: http://dx.doi.org/10.1186/1471-227X-10-17.
4. van Camp LA, Delooz H. Current trauma scoring systems and their applications: a review. In: Trauma Operative Procedures. Topics in Anaesthesia and Critical Care. Springer; 1999. p. 9–29.
5. Wolfgang FD. Research and Uniform Reporting. In: Soreide E, Grande CM, editors. Prehospital Trauma Care. CRC Press; 2001. p. 131–152.
6. Sasser S, Varghese M, Kellermann A, Lormand J. Prehospital trauma care systems. World Health Organization. 2005;.
7. Linn S, Knoller N, Giligan CG, Dreifus U. The sky is a limit: Errors in prehospital diagnosis by flight physicians. The American Journal of Emergency Medicine. 1997;15(3):316 – 320.
8. Regel G, Seekamp A, Pohlemann T, Schmidt U, Bauer H, Tscherne H. Does the accident patient need to be protected from the emergency doctor? Der Unfallchirurg. 1998;101(3):160–175.
9. Boyd CR, Tolson MA, Copes WS. Evaluating trauma care: the TRISS method. Journal of Trauma. 1987;27(4).
10. Breiman L. Random Forests. Machine Learning. 2001;45(1):5–32.
11. TRISS - Overview and desktop calculator; 2016. http://www.trauma.org/index.php/main/article/387/.
12. TRISS - Scoring systems for ICU and surgical patients; 2016. http://www.sfar.org/scores2/triss2.html.
13. Kilgo P, Meredith J, Osler T. Incorporating recent advances to make the TRISS approach universally available. Journal of Trauma. 2006;p. 1002–1009.
14. Cox DR. The Regression Analysis of Binary Sequences. Journal of the Royal Statistical Society Series B (Methodological). 1958;20(2):pp. 215–242. Available from: http://www.jstor.org/stable/2983890.

15. Schetinin V, Jakaite L, Jakaitis J, Krzanowski WJ. Bayesian Decision Trees for predicting survival of patients: A study on the US National Trauma Data Bank. Computer Methods and Programs in Biomedicine. 2013;111(3):602–612.
16. Rughani AI, Dumont TM, Lu Z, Bongard J, Horgan MA, Penar PL, et al. Use of an artificial neural network to predict head injury outcome. Journal of Neurosurgery. 2010;113(3):585–590.
17. Mitchell TM. Machine learning. McGraw Hill series in computer science. McGraw-Hill; 1997.
18. Haykin S. Neural Networks: A Comprehensive Foundation. 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR; 1998.
19. Rumelhart DE, Hinton GE, Williams RJ. Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1. MIT Press; 1986. p. 318–362.
20. Quinlan R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers; 1993.
21. Frank E, Witten IH. Generating Accurate Rule Sets Without Global Optimization. In: ML. Morgan Kaufmann; 1998. p. 144–151.
22. Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician. 1992;46(3):175–185.
23. le Cessie S, van Houwelingen JC. Ridge Estimators in Logistic Regression. Applied Statistics. 1992;41(1):191–201.
24. Kologlu M, Elker D, Altun H, Sayek I. Validation of MPI and PIA II in two different groups of patients with secondary peritonitis. Hepato-gastroenterology. 2000;48(37):147–151.
25. Marshall JC, Cook DJ, Christou NV, Bernard GR, Sprung CL, Sibbald WJ. Multiple organ dysfunction score: a reliable descriptor of a complex clinical outcome. Critical care medicine. 1995;23(10):1638–1652.
26. Biondo S, Ramos E, Deiros M, Ragué JM, De Oca J, Moreno P, et al. Prognostic factors for mortality in left colonic peritonitis: a new scoring system. Journal of the American College of Surgeons. 2000;191(6):635–642.
27. Le Gall JR, Lemeshow S, Saulnier F. A new simplified acute physiology score (SAPS II) based on a European/North American multicenter study. Jama. 1993;270(24):2957–2963.
28. John GH, Langley P. Estimating Continuous Distributions in Bayesian Classifiers. In: UAI. Morgan Kaufmann; 1995. p. 338–345.
29. Platt J. Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: Advances in Kernel Methods - Support Vector Learning. MIT Press; 1998. .
30. Keerthi SS, Shevade SK, Bhattacharyya C, Murthy KRK. Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation. 2001;13(3):637–649.
31. Hastie T, Tibshirani R. Classification by Pairwise Coupling. In: NIPS. vol. 10. MIT Press; 1998. p. 507–513.
32. Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, et al. Top 10 algorithms in data mining. Knowl Inf Syst. 2008;14(1):1–37.
33. Entezari-Maleki R, Rezaei A, Minaei-Bidgoli B. Comparison of Classification Methods Based on the Type of Attributes and Sample Size. JCIT. 2009;4(3):94–102.
34. Committee on Trauma ACoS. NTDB User Manual v. 7.2. https://www.facs.org/quality-programs/trauma/ntdb/datasets; 2009.
35. for Prognosis IIM, of Clinical Trials in TBI A. Injury / Disease Related Events. http://www.tbi-impact.org/cde/; 2010.
36. Arlot S, Celisse A. A survey of cross-validation procedures for model selection. Statistics Surveys. 2010;4:40–79.
37. Fawcett T. ROC Graphs: Notes and Practical Considerations for Researchers. HP Laboratories; 2004.
38. Carletta J. Assessing agreement on classification tasks: the kappa statistic. Computational linguistics. 1996;22(2):249–254.
39. Hall M, Frank E, Holmes G, Pfahringer B, Reutemann P, Witten IH. The WEKA data mining software: an update. SIGKDD Explor Newsl. 2009 Nov;p. 10–18.
40. Cawley GC, Talbot NLC. On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. Journal of Machine Learning Research. 2010;11:2079–2107.
41. Platt J. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In: Advances in Large Margin Classifiers. MIT Press; 1999. p. 61–74.
42. Youden WJ. Index for Rating Diagnostic Tests. Cancer. 1950;3(1):32–35.
43. Hosmer DW, Lemeshow S. Applied Logistic Regression. Applied Logistic Regression. Wiley; 2004. Available from: https://books.google.com.br/books?id=Po0RLQ7USIMC.
44. Hosmer-Lemeshow Statistic;. https://en.wikipedia.org/wiki/Hosmer%E2%80%93Lemeshow_test.
45. Zhu X, Davidson I. Knowledge Discovery and Data Mining: Challenges and Realities . Hershey, PA, USA: IGI Global; 2007.
46. Palmer C. Major trauma and the injury severity score - where should we set the bar? Annals of Advances in Automotive Medicine. 2007;19:13 – 29.
47. Java Runtime Environment;. https://java.com/en/.
48. Hunter A, Kennedy L, Henry J, Ferguson I. Application of neural networks and sensitivity analysis to improved prediction of trauma survival. Computer Methods and Programs in Biomedicine. 2000;62(1):11–19.
49. Silva A, Cortez P, Santos MF, Gomes L, Neves J. Mortality Assessment in Intensive Care Units via Adverse Events Using Artificial Neural Networks. Artif Intell Med. 2006;36(3):223–234.
50. S DC, S AS. A Data Mining Approach for Prediction of Heart Disease Using Neural Networks. IJCET. 2012;3(3):30 – 40.
51. Larson J, Newman F. An Implementation of Scatter Search to Train Neural Networks for Brain Lesion Recognition. Journal of Mathematics. 2011;4(3):203–211.
52. Kim S, Kim W, Park RW. A Comparison of Intensive Care Unit Mortality Prediction Models through the Use of Data Mining Techniques. Healthcare Informatics Research. 2015;17(4):232–243.
53. Corporation I. SPSS Software; 2013. http://www-01.ibm.com/software/analytics/spss/.
54. Michelle S, Hari R, Bryan C, Dua A, Del Junco D, Wade C, et al. Prehospital triage of trauma patients using the Random Forest computer algorithm. Journal of Surgical Research. 2013;187:371–376.
55. Patil BM, Joshi RC, Toshniwal D, Biradar S. A New Approach: Role of Data Mining in Prediction of Survival of Burn Patients. J Medical Systems. 2011;35(6):1531–1542.
56. Lefering R, Huber-Wagner S, Nienaber U, Maegele M, Bouillon B. Update of the trauma risk adjustment model of the TraumaRegister DGU^{TM}: the Revised Injury Severity Classification, version II. Critical Care. 2014;18(5):476. Available from: http://dx.doi.org/10.1186/s13054-014-0476-2.
57. Restrepo-Álvarez CA, Valderrama-Molina CO, Giraldo-Ramírez N, Constain-Franco A, Puerta A, León AL, et al. Trauma severity scores. Colombian Journal of Anesthesiology. 2016;44(4):317 – 323.
58. Amadini R, Sefrioui I, Mauro J, Gabbrielli M. Fast Post-Disaster Emergency Vehicle Scheduling. In: DCAI. vol. 217 of Advances in Intelligent Systems and Computing. Springer; 2013. p. 219–226.