

# Predictability of Kidney Dialysis Survivability

Arvinte Rareș, Diana Trandabăț

<sup>a</sup> *Alexandru Ioan Cuza University, Faculty of Computer Science, Bulevardul Carol I, Nr.11, 700506, Iași, Romania*

## Abstract

In medicine, vast amount of data is stored detailing the actions undertaken for each patient, data very rarely used by hospitals, mostly having statistical and informative purposes for doctors and hospital management. Such data could be successfully used to monitor a patient more accurately, both outside and inside the hospital. Although there exist internal monitoring processes, they can be expensive or limited to actions that are inherited from simple, public statistics, when in fact increased efficiency could be seen if using more complex data. In this paper, we use patient data to predict kidney failure, a work which may inspire others to look for and find patterns in medical health records.

## Keywords <sup>1</sup>

Neural network; Kidney; dialysis; survivability

## 1. Introduction

The human body is a miracle of nature. It consists of the skeleton, muscles and organs. But what happens when an organ doesn't work? Specifically, when the kidneys no longer function properly? Nowadays, modern medicine found a solution to replace its function to help people stay alive because kidneys have the critical mission of cleansing human blood. This method is dialysis which, as specified above, has the role of taking over the mission of the kidneys.

Its role is to eliminate toxins from the blood and it works as follows: the patient's blood passes through a filter that contains a fluid used in dialysis to stop toxins and at the exit the blood is cleansed. The motivation behind choosing this topic is the fundamental idea of helping other people and this came to light when a doctor from Iasi proposed to help the medical community through an application or study in helping people on dialysis, a topic of increased interest in the last years [1]. This way, doctors can help patients better and increase their rate to survive.

This application should help a doctor to make better decisions about the patient's health in the context of dialysis. Through the patient's analyzes, it will be possible to generate graphs to keep the doctor up to date with a patient's health condition. This will be possible with the help of a neural network that is trained with data about dialysis patients, method already studied in the literature [2].

In this paper we will present the application and explain its applicability in society and in future projects. We will also follow the exposition of the technologies used in its creation and possible development ideas in the future but also the description of the stages and difficulties encountered until reaching a final form of the application. By describing the application, we will follow the detailing of its functionalities, the technologies used in the open-source environment and the programming language used, namely Python [3].

The application works on the premise of creating a neural network that is trained and classifies dialysis patients. After if there is a data set of a single patient who presents his long term analyzes can be made an accuracy of his evolution. In the future, the application can incorporate also graphic support and various statistics for the doctor, as suggested in [4].

---

IDDM-2022: 5th International Conference on Informatics & Data-Driven Medicine, November 18–20, 2022, Lyon, France

EMAIL: [raresarvl@gmail.com](mailto:raresarvl@gmail.com); [diana.trandabat@gmail.com](mailto:diana.trandabat@gmail.com)

ORCID: 0000-0002-8494-4441; 0000-0002-9874-0642;



© 2022 Copyright for this paper by its authors.  
Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).  
CEUR Workshop Proceedings (CEUR-WS.org)

## 2. State of the art

Dialysis removes residues and fluids from the body that the kidneys are unable to eliminate. Dialysis also has the role of maintaining the body's balance, correcting the levels of various toxic substances in the blood. Without dialysis, all patients with complete renal impairment would die from the accumulation of toxins in the bloodstream [5].

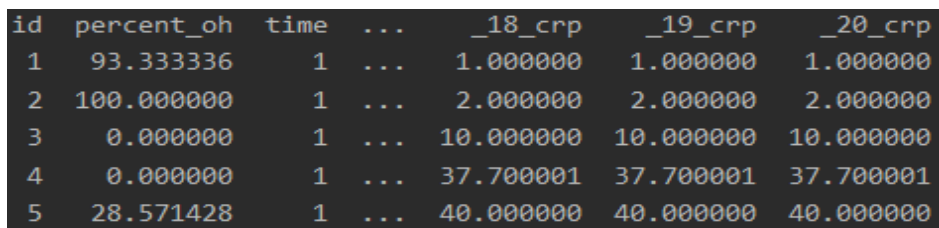
After consulting [6] we concluded that a neural network would be the most appropriate solution. This paper presents the behavior of a neural network, a decision tree, and a logistic regression on a sample of 193 patients undergoing hemodialysis (HD) in Hasheminejad Kidney. Center (HKC) of Tehran, which is one of Iran's largest renal hospitals. This study is similar to mine difference being that they only explain how these algorithms would work using a software framework developed with Tanagra tool which is used for data mining related work. Instead, we built a neural network using tensorflow. It was good because this article is on exactly the same topic as ours, the difference being made by the approach and the data set [7].

The work in [8] addresses the problem from the perspective that vitamin D is the main cause and that people with this deficiency should be identified, prioritizing their monitoring. The authors propose three methods, namely multivariate logistic regression, neural networks and decision trees. As in our dataset, they included issues that would influence decision making such as whether or not the patient is a smoker. Their results on a data set of approximately 900 people indicate that the multivariate logistic regression and neural network have similarly good results to the decision tree.

In the work [9], the authors approach a topic similar to ours, on a different data set, more evenly distributed. Among the characteristics chosen for the neural network, many of them are similar with the ones we use, which makes us lean towards the idea of some obligatory factors that must be considered when it comes to dialysis, indicating that some criteria need to be used with a greater weight than other factors.

## 3. Architecture

The available data set is an .xls file containing the data of 4114 dialysis patients, structured in 240 columns. A sample of the initial data can be seen in Figure 1.

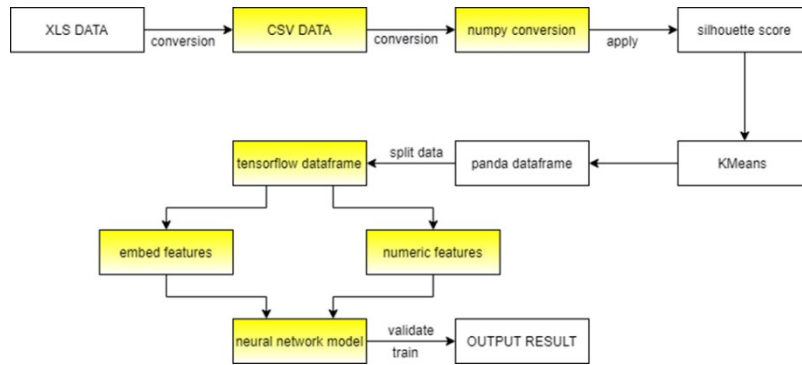


id	percent_oh	time	...	_18_crp	_19_crp	_20_crp
1	93.333336	1	...	1.000000	1.000000	1.000000
2	100.000000	1	...	2.000000	2.000000	2.000000
3	0.000000	1	...	10.000000	10.000000	10.000000
4	0.000000	1	...	37.700001	37.700001	37.700001
5	28.571428	1	...	40.000000	40.000000	40.000000

Figure 1: Data snippet

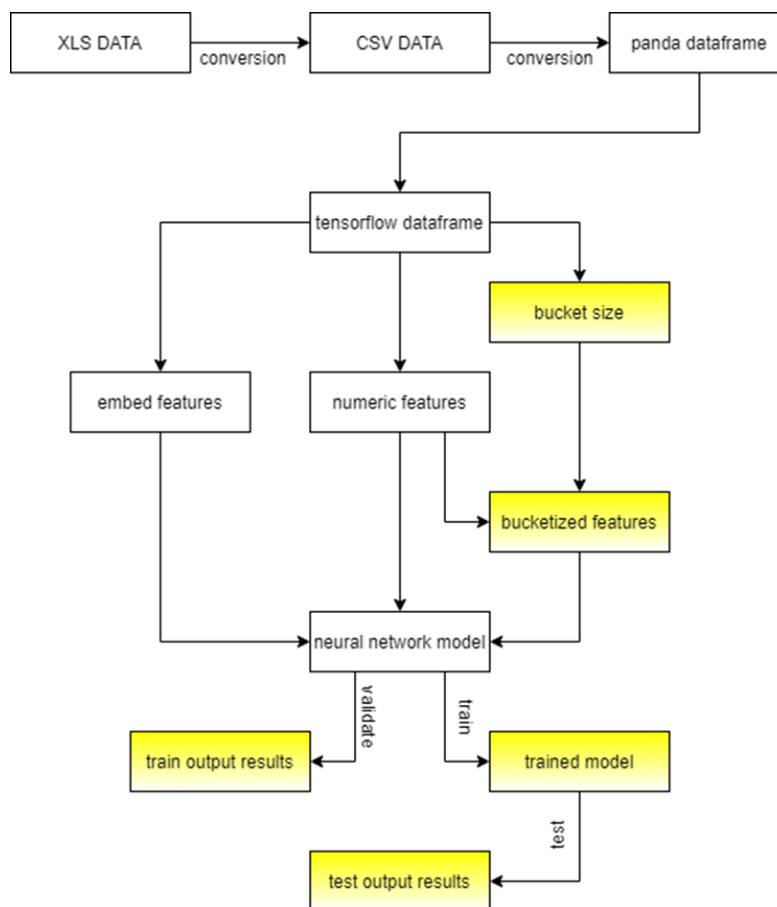
In the first phase, we took these data and processed them in the form of a pandas type dataset. After we tried to create a simple neural network to test the behavior of the data in practice. While running we noticed that some columns contain incomplete data. As an initial solution the missing data were populated with the value 0 but the next problem was the calendar data which indicates when certain types of analyzes were done so we removed them from the data frame. The only problem remains the fact that we did not know which is the output column so we tried to apply the K-means algorithm which, after discussions with a specialist, we had come to the conclusion of placing people in four risk groups.

We applied K-means and each person was assigned a risk group from zero to three. After this grouping we saved it in the data set in a column called "target". Following the passage of this data set with the "target" column as output through the neural network, an accuracy of 43% was obtained. The results do not look so good so we resumed discussions with an expert in the field who suggested cleaning the data, namely from the 240 columns many of them were apparently used to compensate for the missing data that we had filled in with zero at a previous step. The overall architecture can be seen in Figure 2.



**Figure 2:** Data conversion process

After removing duplicate columns, we used the per column average of the values to fill in the missing data. After passing through the neural network, the results increased to 73%, but a closer look revealed that these results varied between 14% and 73%, the reason for this being represented by the vast number of different parameters. Following other discussions with the medical experts, we decided to use the “all\_cause\_death” column as output, and removed all calendar data, except the start and end date of dialysis, which we combined into a single column in which we keep evidence of the number of days the patient undergo dialysis, as suggested in [10]. This column contains values of zero (alive) or one (dead). The results were approximately the same because the number of surviving patients is much higher than the number of the dead ones, the data being highly unbalanced.

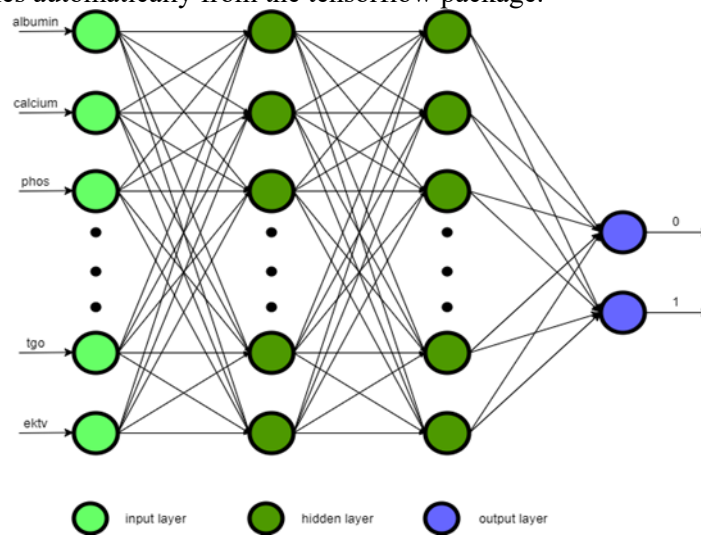


**Figure 3:** Final architecture

We have chosen a method to group the numeric columns in 4 groups, more precisely the numeric columns that do not have values only between zero and three, managing by this to reduce each

numeric class number of possible combinations and to obtain 84% accuracy. Finally, after changing the loss function from categorical cross-entropy to binary cross-entropy, which is better suited for our output, we obtained the final result of 94% accuracy in testing.

The final form of the application works as follows (Figure 3). At first it receives an .csv file at the input which is opened with pandas and automatically converted into a pandas data frame. After that, also with the help of the pandas, the columns with the missing data are filled in with the average per column. Next is the division of the data frame into train data, test and evaluation after these new branches of the initial data frame are converted into tensorflow data frame. The next step is to create the feature layer that serves as input for the neural network. At this step we have three types of features, namely embed, numeric and bucket sized features. Embed is used for columns consisting of strings while numeric is for columns consisting of numbers and bucket sized for certain numeric columns that have been optimized in terms of their input. The network model consists of the first layer which is the input layer and contains all the features followed by two dense layers with relu activation. Then an output layer with sigmoid activation because the output is binary. The model is compiled with the loss function of binary cross-entropy and *adam* optimization with the function  $lr = 1e - 3$  this comes automatically from the tensorflow package.



**Figure 4:** Neural Network architecture

Finally, the model (Figure 4) is fitted and then the test data are executed. In the end, graphs are generated for the analysis of the results and also a method available at the end to test with a patient his medical evolution based on many analyzes created manually.

#### 4. Data description

The initial data set was an XLS file with 4,114 patients and their analyzes along with the dialysis period and whether the person died or not. There are 240 columns of which for the columns “hb”, “ektv”, “calcium”, “phos”, “tgo”, “albumin”, “crp” have rows whose data are missing or are incomplete so for these columns initially there are another 20 identical columns in which an average of the values is applied to fill in these missing values but we considered it necessary to eliminate these duplicated columns and to complete the incomplete or missing data with the average per column, thus reaching the remaining 100 columns. Then there were columns that indicated when the respective medical analysis was done so we eliminated those and finally we joined the start and end date of dialysis in a single column that keeps track of the number of days on dialysis. Thus, the final data set will be 4114 patients with 83 features in an .csv file. From the beginning the data were anonymous, patients being represented by an id column that was finally removed because it does not help anything.

The data respects the principles of GDPR (general data protection regulation), there are also columns that represent the id of the hospital which in the end were deleted as well. Figure 5 shows several data distribution reports to present an overview of the data used. After consultation with a healthcare professional, all remaining features are essential in classifying patients. In the initial data set the features can be divided into five categories, namely demographic, clinical, laboratory data, laboratory data to compensate for missing values and situational. And in the final data set only three categories remain, namely demographic, clinical, laboratory data. Demographic traits are traits that represent aspects of one's appearance. These include traits such as the patient's weight or age. Following are clinical features that are in principle those related to the patient's condition such as whether he suffers from certain comorbidities or has certain addictions such as smoking. Laboratory data and laboratory data to compensate for missing values are the values of patient analyzes. Here we have features from the spectrum of several types of tests from blood to dialysis, some examples would be the level of fat-mass, the number of white blood cells, calcium or albumin levels. This is where most of the features are included, with the largest share in the decision-making factor. Finally, situational traits are related to identification data such as the patient's ID or date of birth, which in the final data are no longer taken into account and do not present a factor for making a decision.

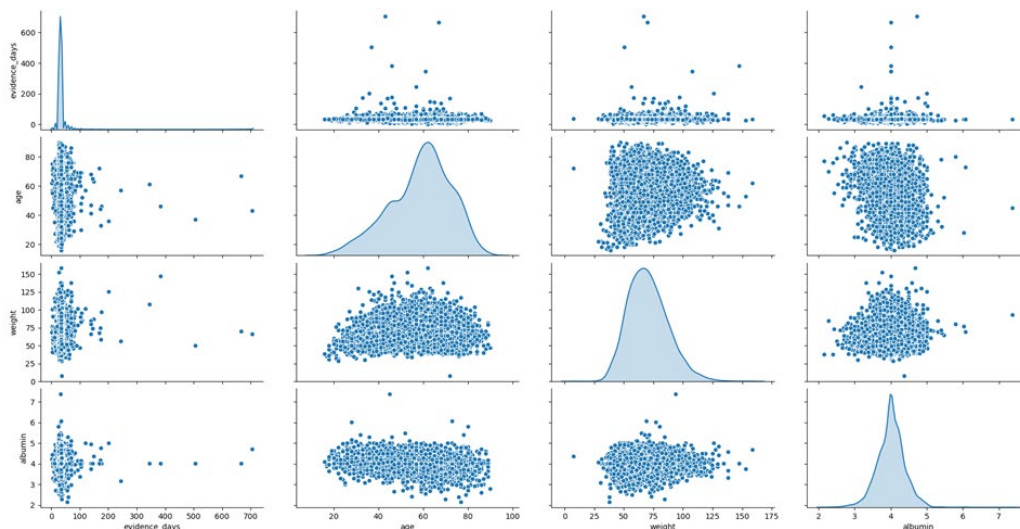


Figure 5: Statistic data

## 5. Modules

The application is divided into several modules to clean the data and prepare the neural network to be executed successfully. It does not have a graphical interface being only taken to the test stage. Below we describe the modules and how they work to understand why we chose this division.

*Data frame reader module:* The first module is to read the data frames according to the required situation such as the file format. For this we use the pandas library which provides functions for reading xls files or csv files. This module, in addition to reading the file and converting to a panda dataframe, also deals with filling in the missing data with the media on the column. This is also done with pandas.

*Conversions module:* This module deals with the conversion of data, columns or data frames into new data types. There are three functions in this module. The first function is to convert an array of strings of a pandas data frame into one of numbers of a pandas data frame. This function receives as input the data frame, the string list, the column name and the last index. This function can be used for any column that contains strings but for our data it had to be used only for the "renal\_diagnosis\_icd" column. Also, this function is used only to be able to do k-means clustering because in this method we use only numpy data (only accept numbers). At output we have a copy of the data frame with changes done. The next function is a simple one that receives a pandas data frame and converts it into a numpy

object. And this function is used only for k-means that act as input for it. The final function is the most important in the module, it receives a pandas data frame and converts it to tensorflow data set. This function also removes the output column from the data frame because it is not relevant for tensorflow data frame to contain this column. The output of this function return a tensorflow data frame that will represent the input for the neural network.

*Miscellaneous module:* This module contains three functions the first is to extract all the elements of a column from the data frame. It receives as input the data frame and the column name and at the output we have a list with all the elements of the column. This function is used only for the "renal\_diagnosis\_icd" column. The next function is to create a list that contains a number of equal intervals used to optimize the neural network. This function receives the elements of a numeric column and the number of intervals, calculates the length of an interval and saves in a list the beginning of each interval. The last function receives the date frame as input and returns a list of all the column names that will later be used to divide the columns into feature types.

*K-means module:* There is a dedicated module for k-means that contains the whole process of performing a clustering with this method. It contains two functions the first of which calculates the silhouette score which is used to decide the number of clusters, it uses the sklearn library to calculate this score. It receives as input the data frame and the number of iterations (clusters) and executes k-means for each number of clusters and for each score it is checked if it is higher than the maximum found. If it is higher it becomes the new maximum and the iteration number becomes the number of clusters. The other function is actually the k-means clustering used with the help of sklearn. In the end its also plot a graph with all its points and centroid.

*K-means executor module:* This module handles the whole process of calling all the functions needed to read, convert and execute the cluster.

*Neural network module:* The neural network module contains the whole process of creating the features (from tensorflow data frame) and the model that will be executed in the execution module. With the help of the tensorflow library, a division is made into features that will be used as the first layer of the neural network. Three types of features are used, the first being embed feature with the only column of this type being "re-nal\_diagnosis\_icd". Then numeric features that are divided into simple and bucketsized features. Here we use the interval division method described above to create these buckets. Simple numeric features are those that contain integer data in the range of zero - three. This condition was chosen ad hoc after manual data analysis. The rest of the data is of the bucket features type. After these they are used as the first layer of the neural network followed by two dense layers and a dense layer used as output. The compilation is done with the Adam optimizer using the binary cross-entropy loss function. It also contains a function for creating the neural network model that will receive these features created above. And at the end of the module is a test function used to test the system with a new patient with multiple analyzes.

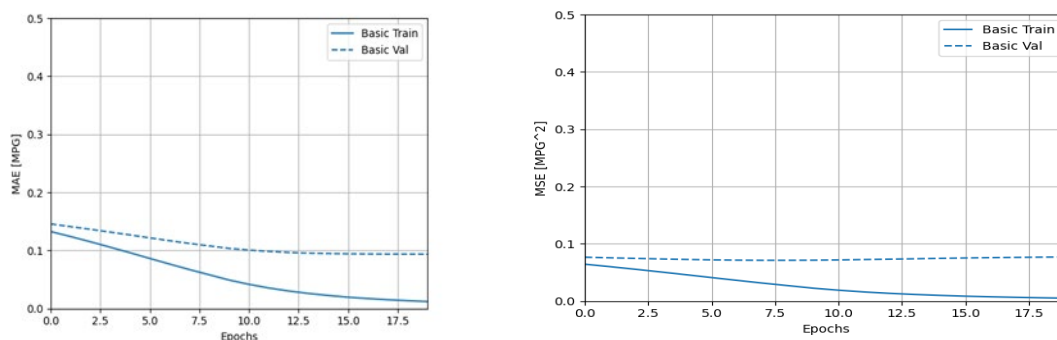
*Execution module:* The execution module contains the whole process of executing the neural network and graph creation. Here the train, test and evaluated data are converted from pandas data frame to tensorflow data frame using the conversion module. Also here is the training of the neural network, its evaluation and testing. Here the statistical functions from the statistics module are called, but also here the execution of the testing of a new patient is called.

*Statistics module:* This module deals with any means of creating graphs or displaying statistics. It contains functions for example the creation of statistics on the test data following the evaluation of the system or the evolution of the patient over time or the confusion matrix.

*File manipulation module:* This module deals with manipulating the data to bring it into an ad-hoc final form of our choice. It contains functions for converting a file from xls to csv or creating copies of a file with the addition of a new column or even converting columns that contain time periods into numbers or merging two time columns into a numeric column. This functions were used in the early stages of the project when we used centroid cluster clustering as output. At the end of the project it is no longer used because the "all\_cause\_death" column is used as output.

## 6. Results

The factors considered for evaluating the results consisted in a number of metrics: loss, performance classification, accuracy, precision, recall, AUC and F-score. After training the neural network and validating it with the validation data, the results obtained can be interpreted from Figure 6.



**Figure 6:** MAE graph and MSE graph

The first image shows the evolution of mean absolute error (MAE) which helps us to get an overview of how the error rate evolves with each epoch. If the line becomes constant it means that the neural network is no longer learning. The MAE score after 20 epochs is 0.074.

The other image shows the evolution of mean squared error (MSE) which evaluates the quality of predictions. The closer it is to zero, the better the quality. It is observed that the value after 20 epochs is close to zero. The result value is 0.067. After training we tested the system with test data (20% of the total set date). Accuracy is a metric for measuring how often the neural network correctly classifies an input from the given set date.

**Table 1**

Metrics

Metric	Values
Accuracy	0.929
Precision	0.870
Recall	0.641
AUC	0.843
MAE	0.068
MSE	0.077
F-score	0.738

As can be seen in Table 1, the accuracy obtained is 92.9%, which indicates a fairly good score. The precision and recall are calculated as follows: precision is the total of true positives (TP) divided by the sum of true positives and false positives (FP) and the recall is the total of true positives divided to the sum of the total of true positives and false negatives (FN). Precision shows us the probability that a patient is classified correctly with one of the two values while the recall indicates the probability that a patient is cataloged with a certain tag. After testing the neural network, the precision obtained was 87% and a recall of 64%.

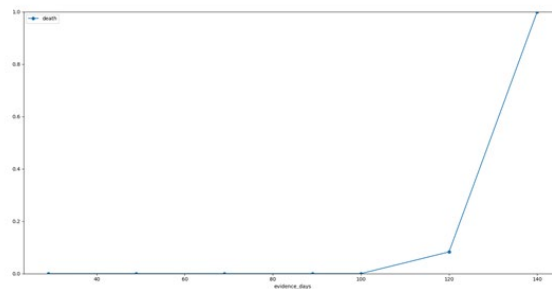
F-score its weighted harmonic mean helps us to have an overview of the accuracy of the test data. Its value is between zero and one. The closer it is to one we can say that the system does a better job if new data comes at the input. The F-score obtained after the test was 0.738 which indicates that the system is doing quite well if would receive new data as input.

ROC or receiver operating characteristic curve is a graph that shows the performance of the classifier in the ratio true positive rate (TPR) versus the false positive rate (FPR) [11].

Above we talked about ROC to lay a foundation in explaining the AUC. AUC (area under the ROC Curve) as the name implies refers to the area under the graph which tells us how well the neural network distinguishes between the classes of a classifier, in our case it shows how well it distinguishes between living or dead patients. The AUC value is between zero and one and the closer it is to one the better the system makes the differences. The AUC after testing was 0.843.

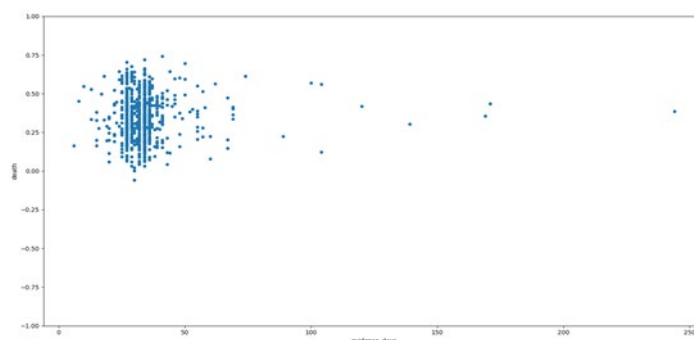
Confusion matrix helps to understand how classified data is distributed so that it is understood how well the system handles. Out of the total number of 4114 patients, 823 are test patients. Following the evaluation, 682 were classified as true negatives, 82 true positives, 13 false positives, 46 false negatives.

As can be seen, the data are quite unbalanced because many patients are classified as alive or true negatives in our case. In the first instance this was a problem but in the end after we applied methods to stop this by balancing the values of the features we managed to ignore this and therefore the program makes few mistakes. After testing the system, we decided to check his behavior for a new patient with multiple tests and to perform an evolution of his disease. Figure 7 shows the evolution of the patient's disease depending on the number of days since he is on dialysis, thus demonstrating how the patient's risk of dying over time can be interpreted.



**Figure 7:** Patient evolution in time

Figure 8 shows the distribution of patients according to the time spent on dialysis in relation to their near death. This figure indicates how the neural network classifies patients according to time so one can see the results of this approach.



**Figure 8:** Results in comparison to evidence days

Comparing our results to the ones discussed in the state of the art sections, we can infer, for example, that in [6] the aim of the paper is to compare several data mining techniques to predict whether a person would survive kidney dialysis. Because we only approached the neural network method, we can only compare with their neural network method. First of all, it should be noted that their data set is small, balanced and has only eight parameters taken into account. They also used the Tanagra tool which provides a suite of tools for the mining data. Like us, they used the accuracy to determine the performance, their score being 93.852% compared to the one we obtained, namely



92.9%, but compared to them we did not average all the runs (the accuracy varies between 91% and 96%) this variation is due to the fact that our data is more balanced than their data.

The paper [9] addresses neural networks and logistic regression, the comparison metric chosen for comparison is AUC. We will also refer only to the results obtained on the neural network. They used Neuroshell 2 version 3.0 to make the network, on a data set of 3629 dialysis patients and got an AUC of 0.7602 compared to our neural network which got an AUC of 0.8437. Our opinion is that this result reflects the optimizations made for bucket sized features.

## 7. Conclusions and further work

Above we discussed the problem addressed and the solution we chose, namely a way to predict the evolution of a patient on dialysis based on his probability of dying. We presented the architecture of the neural network built using python and tensorflow and the results obtained by it. Our conclusion is that this issue has been addressed in the past and we agree with the opinion of other authors who have concluded that neural networks have the best results [12].

The difference brought by this paper is the data set tested and the results obtained. We could say that we have improved the results obtained by them. we believe that computer science should have a greater impact on such areas that work with human life, not to replace decision-making but to be introduced as an assistant for it thus increasing the speed of response and consideration of opinions which at the moment may have escaped for example a doctor.

Also, what we have done can be seen as a starting point for the modernization of medical infrastructures in Romania, it can help to develop applications for the benefit of doctors, thus helping the community and saving more lives. It should be added in conclusion that this application was made in python with tensorflow and not with an online application that provides tools to make this possible

The application is purely demonstrative, for the future we will try to create a graphical interface and a database in which to store future patients so that it can be used in hospitals. Not to replace doctors but to help them make a decision taking into account the choices made by the program. Thus, the program can show a doctor the evolution of the patient and he can act as such. The program can also interpret for the doctor the data provided so as to help him even more, highlighting the features that have changed in the patient and where to work with him, the benefits provided as a recommendation because in the end something subjective must to make the decision to solve the problem is not an objective machine [13]. Also, if this application is to be further developed in the future, it can motivate people to try new methods and improve the system, thus increasing the chance of success and to be able to be introduced in hospitals to help save human lives.

## 8. References

- [1] Tomašev, N., Glorot, X., Rae, J. W., Zielinski, M., Askham, H., Saraiva, A., ... & Connell, A. (2019). A clinically applicable approach to continuous prediction of future acute kidney injury. *Nature*, 572(7767), 116-119.
- [2] Chan, L., Vaid, A., & Nadkarni, G. N. (2020). Applications of machine learning methods in kidney disease: hope or hype?. *Current opinion in nephrology and hypertension*, 29(3), 319.
- [3] Van Rossum, G. (2007, June). Python Programming Language. In USENIX annual technical conference (Vol. 41, p. 36).
- [4] Kim, H. W., Heo, S. J., Kim, J. Y., Kim, A., Nam, C. M., & Kim, B. S. (2021). Dialysis adequacy predictions using a machine learning method. *Scientific reports*, 11(1), 1-7.
- [5] Thongprayoon, C., Hansrivijit, P., Bathini, T., Vallabhajosyula, S., Mekraksakit, P., Kaewput, W., & Cheungpasitporn, W. (2020). Predicting acute kidney injury after cardiac surgery by machine learning approaches. *Journal of Clinical Medicine*, 9(6), 1767.
- [6] Lakshmi, K. R., Nagesh, Y., & Krishna, M. V. (2014). Performance comparison of three data mining techniques for predicting kidney dialysis survivability. *International Journal of Advances in Engineering & Technology*, 7(1), 242

- [7] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Kudlur, M. (2016). Tensorflow: A system for large-scale machine learning. In 12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16) (pp. 265-283).
- [8] Bhan, I., Burnett-Bowie, S. A. M., Ye, J., Tonelli, M., & Thadhani, R. (2010). Clinical measures identify vitamin D deficiency in dialysis. *Clinical Journal of the American Society of Nephrology*, 5(3), 460-467.
- [9] Tangri, N., Ansell, D., & Naimark, D. (2008). Predicting technique survival in peritoneal dialysis patients: comparing artificial neural networks and logistic regression. *Nephrology Dialysis Transplantation*, 23(9), 2972-2981.
- [10] Verplancke, T., Van Looy, S., Steurbaut, K., Benoit, D., De Turck, F., De Moor, G., & Decruyenaere, J. (2010). A novel time series analysis approach for prediction of dialysis in critically ill patients using echo-state networks. *BMC medical informatics and decision making*, 10(1), 4.
- [11] Flach, P. A. (2003). The geometry of ROC space: understanding machine learning metrics through ROC isometrics. In *Proceedings of the 20th international conference on machine learning (ICML-03)* (pp. 194-201).
- [12] Cassol, C., & Sharma, S. (2021). Nephrology Lagging Behind in Machine Learning Utilization. *Kidney Medicine*, 3(5), 693-695.
- [13] Zhou, X. J., Zhong, X. H., & Duan, L. X. (2022). Integration of Artificial Intelligence And Multi-omics in Kidney Diseases. *Fundamental Research*.