Using Donor Management Goals Information to forecast Organ Outcomes by using Machine Learning

Rolando Retana

Software Development Department, Transplant Connect Inc

rretana@transplantconnect.com

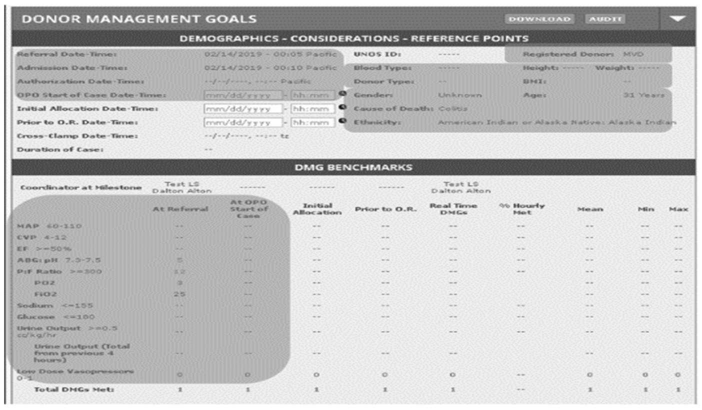
Abstract- Donor Management Goals subsystem (DMG) contains non-traceable information related to the organ recovery process that is entered on the iTransplant system. This information is organized in a timeline and it traces the organ recovery process before its starts to the end. It does keep the information such as temperature, medications used, urine analysis, blood, gases, demographics, etc. All that data in a timeline that brings information to the team before, during and after the organ recovery. In this research this information is intended to be used to forecast the organ outcomes before the recovery of the organs happens.

Index Terms- DMG, Donor Management Goals, Machine Learning, Tensors, TensorFlow.

I. Introduction

Since 2012 multiple OPOs have been contributing to the National DMG Registry and until 2019 UNOS DMG had 17 OPOs members. Several of these members use the iTransplant System to collect the DMG data.

In iTransplant the page DMG is used by several of the OPOs to track their goals and then in several of the cases submit this information to the UNOS DMG Registry. The information on the subsystem DMG is read only and it comes from different modules and pages of the iTransplant System.



DONOR MANAGEMENT GOALS

DEMOGRAPHICS - CONSIDERATIONS - REFERENCE POINTS

Referral Date-Time:	02/14/2019 - 00:05 Pacific	UNOS ID:	-----	Registered Donor:	MVD
Admission Date-Time:	02/14/2019 - 00:10 Pacific	Blood Type:	-----	Height: -----	Weight: -----
Authorization Date-Time:	--/--/----, --:-- Pacific	Donor Type:	--	BMI:	--
OPO Start of Case Date-Time:	mm/dd/yyyy - hh:mm	Gender:	Unknown	Age:	31 Years
Initial Allocation Date-Time:	mm/dd/yyyy - hh:mm	Cause of Death:	Colitis		
Prior to O.R. Date-Time:	mm/dd/yyyy - hh:mm	Ethnicity:	American Indian or Alaska Native: Alaska Indian		
Cross-Clamp Date-Time:	--/--/----, --:-- tz				
Duration of Case:	--				

DMG BENCHMARKS

Coordinator at Milestone	Test LS Dalton Alton	------	------	------	Test LS Dalton Alton				
	At Referral	At OPO Start of Case	Initial Allocation	Prior to O.R.	Real Time DMGs	% Hourly Met	Mean	Min	Max
MAP 60-110	--	--	--	--	--	--	--	--	--
CVP 4-12	--	--	--	--	--	--	--	--	--
EF >=50%	--	--	--	--	--	--	--	--	--
ABG: pH 7.3-7.5	5	--	--	--	--	--	--	--	--
P:F Ratio >=300	12	--	--	--	--	--	--	--	--
PO2	3	--	--	--	--		--	--	--
FiO2	25	--	--	--	--		--	--	--
Sodium <=155	--	--	--	--	--	--	--	--	--
Glucose <=180	--	--	--	--	--	--	--	--	--
Urine Output >=0.5 cc/kg/hr	--	--	--	--	--	--	--	--	--
Urine Output (Total from previous 4 hours)	--	--	--	--	--		--	--	--
Low Dose Vasopressors 0-1	0	0	0	0	0	--	0	0	0
Total DMGs Met:	1	1	1	1	1	--	1	1	1

Figure 1 DMG Subsystem implemented on iTransplant.

The first two columns of information in the DMG Benchmarks sections are expected to be entered before the actual organ recovery happens, this is the data used to generate a forecasting.

By modifying the DMG subsystem/page to provide multiple anonymous records from multiple cases as a data set, and then filter each of the rows to leave only the ones that could provide useful insight by removing rows that contain too many null values, incomplete information or clear mistakes by the user when entering the information.

The collected information as displayed in figure 3, will be used to create a tensor model or a Neural network for machine learning.

Then using a logistic regression mathematical model and the data collected it will forecast in a scale from 0 to 1 the probability of being a successful recovery for heart, liver, kidneys.

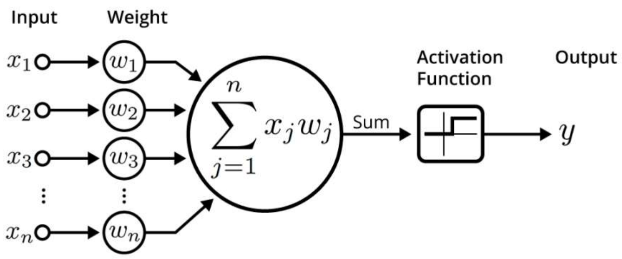


Figure 2 Neural unit used by the tensor. Each unit will be connected to others in a network where the output of one layer becomes the input for the next layer until it reaches the last layer and generate the forecast.

Then using as inputs, the same datapoints that were used to train the model, it should provide as a result an estimated probability forecasting of the outcome being a positive organ recovery or not to a specific organ type.

II. Data Gathering

After analyzing multiple databases in production with the information as of March 2019, not many partners were using the DMG page to gather the information despite that they have it enabled in the iTransplant System.

Many of the OPOs with hundreds of thousands of cases only have a small number of DMG entries.
Initially to determine the cases that had information entered on the DMG the data table called "`DmgInformation`" was used, on multiple OPOs that have this page enabled.
Example:

```
        select COUNT(ID) from DmgInformation
```

After analyzing multiple databases in production with the information as of March 2019, not many partners were using the DMG page to gather the information despite that they have it enabled in the iTransplant System.
Many of the OPOs with hundreds of thousands of cases only have a small number of DMG entries.
Since not many OPO's use the page, it was considered to start with only a few of the clients, for the research we proceed with a few OPO's cases until we collected about 5000 cases.

III. Feature Selection

Looking at the data we have, we can find multiple types, we get dates indicating events, labels indicating categories (blood type, gender, Cause of death, etc), numbers that are indicators of hard data (BMI, weigh, height, etc) This data is collected and adapted to be able to feed the model.

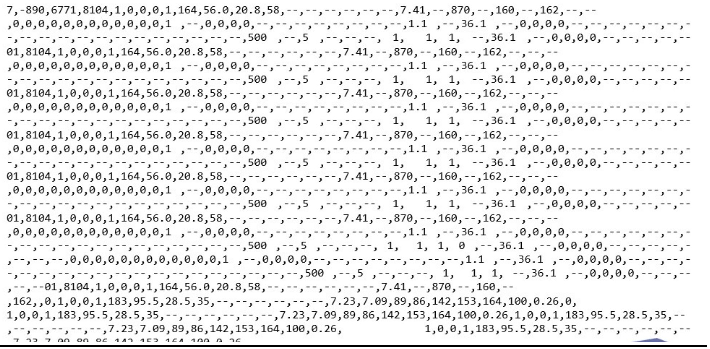
Labels were used to separate some of the data points, two examples of this are *gender* and *cause of death (COD),* where for gender we had in the system only 2 possible values, and for COD we had several possible values but 3 of them repeated over and over and the others were really rare, so it was decided that:

- For gender two columns will be added, "Female" and "Male", when the selected gender was male, the column male would have a 1 and the female a 0, and vice versa. It was expected that there was a 1 or a 0 for both columns.
- For COD, there was a big list of causes of death, but it was considered only the three that were the most common CODs: Anoxia, CVA Stroke and Head Trauma. When the COD was in one of the categories a 1 would be present, but it none of those were the COD then all of them would be set to 0.

Based on the problem and the type of data available, it is decided to continue with a Logistic Regression Model. This will allow to use safely the labels like gender, blood type and COD.

The DMG page in the iTransplant system was modified to iterate among all the cases that contained a record for "`DmgInformation`" and then instead of the regular display on the page, each of the rows was stored in a file, one row per line ignoring any identifiable patient information (names, address, etc) and collecting only information displayed on the demographics, benchmarks and outcomes.

Figure 3 is an example of the data collected; each line represents a record or a case that contained part of the DMG data entered.



```
7,-890,6771,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--,162,--,--
,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,   1, 1,  --,36.1 ,--,0,0,0,0,--,--,--,--,--
01,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--,162,--,--,--
,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,   1, 1,  --,36.1 ,--,0,0,0,0,--,--,--,--,--
01,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--,162,--,--,--
,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,   1, 1,  --,36.1 ,--,0,0,0,0,--,--,--,--,--
01,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--,162,--,--,--
,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,   1, 1,  --,36.1 ,--,0,0,0,0,--,--,--,--,--
01,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--,162,--,--,--
,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,   1, 1,  --,36.1 ,--,0,0,0,0,--,--,--,--,--
01,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--,162,--,--,--
,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,  1, 1, 0 ,--,36.1 ,--,0,0,0,0,--,--,--,--,-
,--,--,--,0,0,0,0,0,0,0,0,0,0,0,0,1 ,--,0,0,0,0,--,--,--,--,--,--,--,--,1.1 ,--,36.1 ,--,0,0,0,0,--,--,--,-
-,--,--,--,--,--,--,--,--,--,--,--,--,--,--,--,500 ,--,5 ,--,--,--, 1,  1, 1, --,36.1 ,--,0,0,0,0,--,--,--
,--,--01,8104,1,0,0,0,1,164,56.0,20.8,58,--,--,--,--,--,--,7.41,--,870,--,160,--
,162,,0,1,0,0,1,183,95.5,28.5,35,--,--,--,--,--,--,7.23,7.09,89,86,142,153,164,100,0.26,0,
1,0,0,1,183,95.5,28.5,35,--,--,--,--,--,--,7.23,7.09,89,86,142,153,164,100,0.26,1,0,0,1,183,95.5,28.5,35,--
,--,--,--,--,--,7.23,7.09,89,86,142,153,164,100,0.26,          1,0,0,1,183,95.5,28.5,35,--,--,--,--,--,--
7.23,7.09,89,86,142,153,164,100,0.26
```

Figure 3 Example of how a file containing the rows of the DMG page looks like. Thousands of rows were collected, not real data, illustration purposes only.

The data collected on each case was:

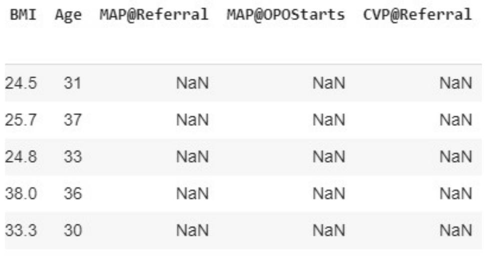
- **Referral Hour**: the time where the referral call was placed, as a reference on when the OPO was aware of the case. This data point was only considered but was not actually collected or included in the dataset, it was only used as a reference for the other date times.
- **Admission Date Time**: minutes passed from the Referral hour until the admission.
- **Authorization Date time**: minutes passed since the Referral hour until the Authorization time.
- **Prior to OR Date Time**: minutes passed from the referral until the patient entered to the OR.

Any of the times can be negative, since the timeline does not precisely follow an order, admission of the patient can happen before the referral or after as well as any other times.

- **Blood Type**: Type of blood, going from 0 to 7, representing the 4 main blood types and their RhD.
- **Donor Type**: Classification between organ, tissue, eye and the combination of them. Using numbers to classify them from 0 to 6. As example: O, T, E, OT, OE, OTE, TE.
- **Female**: 1 when female, 0 when not female.
- **Male**: 1 when male, 0 when not male.
- **Height**: recorded height in centimeters, provided as a number.
- **Weight**: recorded weight in pounds, as a number.
- **BMI**: precalculated BMI.
- **Age**: Number of years recorded for that patient this number can be used in days, weeks etc.
- **MAP at Referral:** Mean arterial pressure when the case was referred.
- **MAP at OPO Starts**: Mean arterial pressure when the OPO recorded it.

- **CVP at Referral:** Central venous pressure when the case was referred.
- **CVP at OPO Starts:** Central venous pressure when the OPO took the case.
- **EF at referral:** Ejection Fraction when the patient was referred.
- **EF at OPO starts:** Ejection fraction when the OPO measured it.
- **P:F** at referral and when OPO starts as a separate inputs.
- **Sodium** level at referral and when the OPO starts.
- **Glucose** at referral and when the OPO starts.
- **Urine** at referral and when OPO starts.
- **Dose** at referral, and when the OPO starts.
- **Dopamine** at referral and when the OPO starts.
- **Neosynephrine** at referral and when the OPO starts.
- **Levophed** at referral and when the OPO starts.
- **Epinephrine** at referral and when the OPO starts.
- **Dobutamine** at referral and when the OPO starts.
- **Creatinine** at referral and when the OPO starts.
- **Levothyroxine** at referral and when OPO starts.
- **Vasopressin** at referral and when the OPO starts.
- **Cardiac Index** at referral and when the OPO starts.
- **Cardiac Output** at referral and when the OPO starts.
- **Stroke Volume Variation** at referral and when OPO starts.
- **Pulse pressure variation** at referral and when the OPO starts the case.
- **Lactate** levels at referral and when the OPO starts the case.
- **Temperature** at referral and when the OPO starts the case.
- **Insulin** total since the last reference (at referral and at the OPO starts).
- **Troponin** at referral and when the OPO starts the case.
- **BNP** at referral and when the OPO starts the case.
- **AST** at referral and when OPO starts the case.
- **ALT** referral/OPO starts.
- **Total Bili** referral/OPO starts.
- **Direct/Conjugated Bili** at referral/ when OPO starts the case.
- **Amylase** at referral and OPO starts the case.
- **Lipase** referral/OPO starts.
- **Ventilation Mode** referral/OPO starts.
- **Ventilation Mode Other** referral/OPO starts.
- **TV** at referral and when OPO starts.
- **Peep** referral/OPO starts.
- **PIP** referral/OPO starts.
- **Liver**: 1 if the outcomes of a liver recovery was successful, 0 when it was not.
- **R Kidney**: 1 when the right kidney was successfully recovered and 0 when it was not.
- **L Kidney**: 1 when left kidney was successfully recovered and 0 when it was not.
- **Heart**: 1 when heart was successfully recovered and 0 when it was not.

The dataset had some rows that contained null values, that had to be filtered out, when many of the columns had null values, the column would be completely removed since it was not useful for forecasting, but when most of the rows had that column value and only a few were null, the rows were the ones eliminated.



BMI	Age	MAP@Referral	MAP@OPOStarts	CVP@Referral
24.5	31	NaN	NaN	NaN
25.7	37	NaN	NaN	NaN
24.8	33	NaN	NaN	NaN
38.0	36	NaN	NaN	NaN
33.3	30	NaN	NaN	NaN

Figure 4 Too many null column values.

Multiple analyses were performed to determine what kind of normalization was needed, and what columns and rows should be kept. Several of the columns mentioned had to be removed and the dataset went from ~60 datapoints to only about 16 datapoints. Now is the time to filter which of these datapoints will become inputs for the neural network.
We are looking to create a parsimonious model that we can train faster and more efficiently; hence the importance of removing redundant or irrelevant inputs is to prevent underfitting or overfitting by affecting too much the lambda [2], for these cases it may be considered to use L1 regularization. Eventually the NN will adjust the weighs to have the irrelevant data

Removing BMI, Height or Weight
Simplicity in a neural network will save a lot of computational time, having BMI in it seems like a redundancy since we already have the values used to calculate it included in the inputs. So, it was decided to have a few experiments and see if it would save or consume additional training time having the values in there or if it would affect the outcome.

Removing gender from the inputs
As we see in the next figure, the number of female donors is lower than the number of male donators, but in a closer look at their pairplots against the outcomes, it seems the percentage of successful recoveries did not vary. The next figure represents the percentage of recovery ratio where 1 is female and 0 is non female (entered as male).

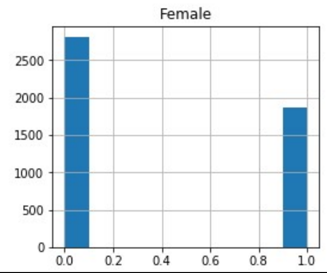


Figure 5. 0 are male donors and 1 are female donors.

The next figure represents the recovery ratio of liver on female and male, there are not labeled to avoid bias, but it is visible that the gender is not visibly relevant on the recovery outcome. One figure represents male and the other represents female.

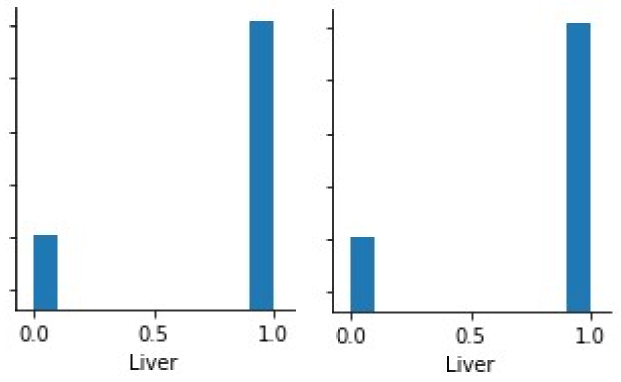
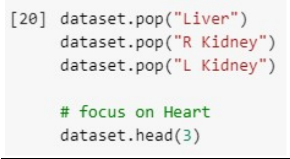


Figure 6 Ratios of successful recovery for liver in men and women. 1 Represents successful recovery and 0 represents not successful recovery.

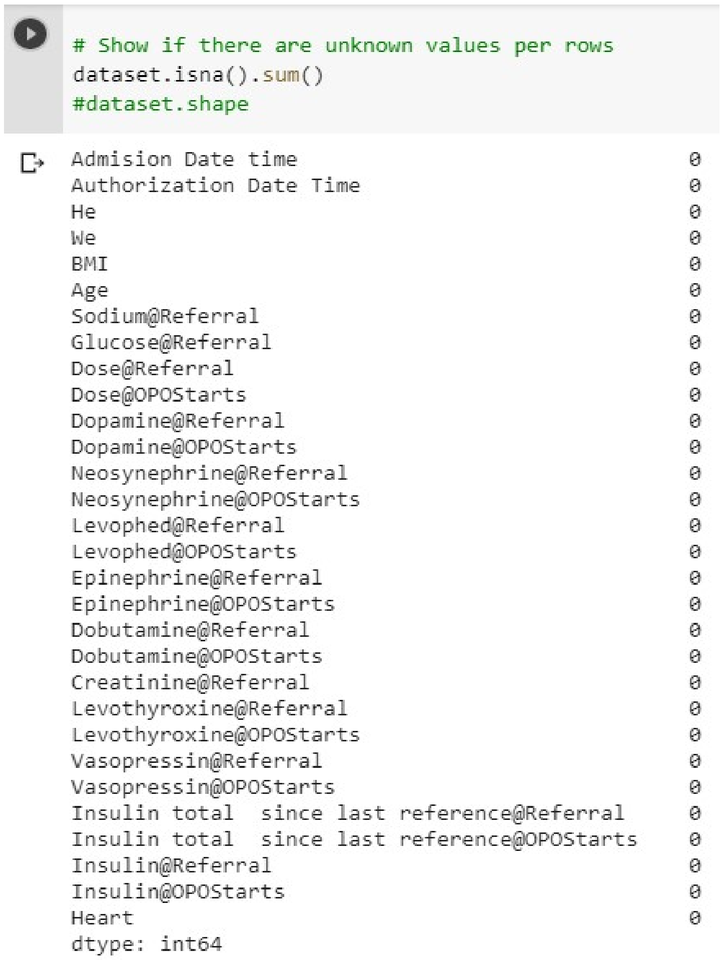
Due to time restrictions the goal of forecasting the successful recovery of heart, liver and kidneys, this experiment will split each organ forecasting separately, since training the model may take exponentially longer than training for forecasting a single organ.



```
[20] dataset.pop("Liver")
     dataset.pop("R Kidney")
     dataset.pop("L Kidney")

     # focus on Heart
     dataset.head(3)
```

Figure 7 Leaving only heart recovery outcomes in the dataset.

The final features included in the learning model are the ones displayed in the next figure, they are the values that have no null remaining.



```
# Show if there are unknown values per rows
dataset.isna().sum()
#dataset.shape
```

```
Admision Date time                                0
Authorization Date Time                           0
He                                                0
We                                                0
BMI                                               0
Age                                               0
Sodium@Referral                                   0
Glucose@Referral                                  0
Dose@Referral                                     0
Dose@OPOStarts                                    0
Dopamine@Referral                                 0
Dopamine@OPOStarts                                0
Neosynephrine@Referral                            0
Neosynephrine@OPOStarts                           0
Levophed@Referral                                 0
Levophed@OPOStarts                                0
Epinephrine@Referral                              0
Epinephrine@OPOStarts                             0
Dobutamine@Referral                               0
Dobutamine@OPOStarts                              0
Creatinine@Referral                               0
Levothyroxine@Referral                            0
Levothyroxine@OPOStarts                           0
Vasopressin@Referral                              0
Vasopressin@OPOStarts                             0
Insulin total  since last reference@Referral      0
Insulin total  since last reference@OPOStarts     0
Insulin@Referral                                  0
Insulin@OPOStarts                                 0
Heart                                             0
dtype: int64
```

Figure 8 Final features included in the dataset.

IV. THE MODEL

A sequential model with three densely connected layers and one output that returns a single continuous value was selected.
The following figure tries to explain the shape of the model selected:

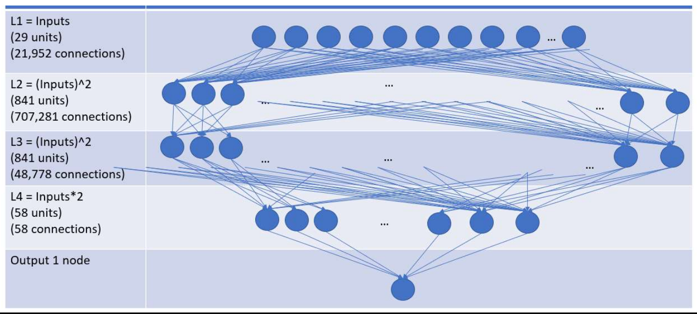
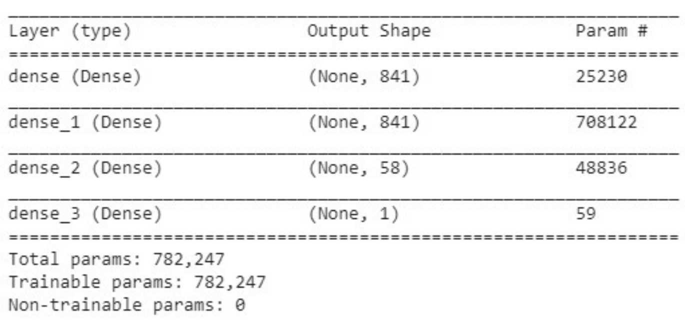


Figure 9 Representation of the learning model.



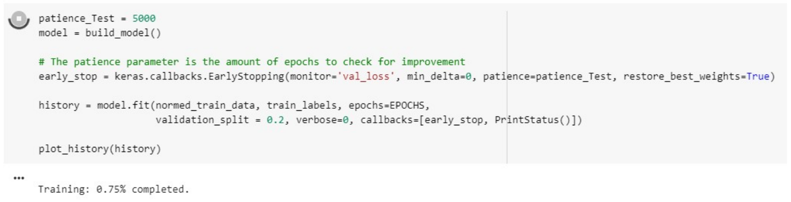
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 841)               25230
_________________________________________________________________
dense_1 (Dense)              (None, 841)               708122
_________________________________________________________________
dense_2 (Dense)              (None, 58)                48836
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 59
=================================================================
Total params: 782,247
Trainable params: 782,247
Non-trainable params: 0
```

Figure 10 Screenshot of Keras representing the learning model in numbers.

The model has a total of 782,247 trainable parameters.

Training the model

It was chosen 20,000 epochs to train it and a callback function was added to print the training status after every 100 epochs.
Running on Google Colab it should take about 4 to 5 hours training the model.
Patience: This variable determines how many epochs can pass without a change before the algorithm stops. This will help stop the training if there is no improvement or change. In order to prevent going the full 4 hours without any improvement until the end, a patience limit was set to 500.
After the patience reach its limit Keras will restore the ones considered as best weights (the ones that had more accurate forecasting).

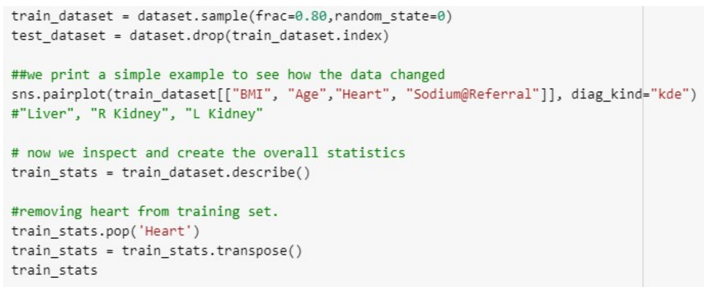


```
patience_Test = 5000
model = build_model()

# The patience parameter is the amount of epochs to check for improvement
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', min_delta=0, patience=patience_Test, restore_best_weights=True)

history = model.fit(normed_train_data, train_labels, epochs=EPOCHS,
                    validation_split = 0.2, verbose=0, callbacks=[early_stop, PrintStatus()])

plot_history(history)

...
Training: 0.75% completed.
```

Figure 11 Piece of code setting up the patience and starting the fitting process.

Train dataset and test dataset

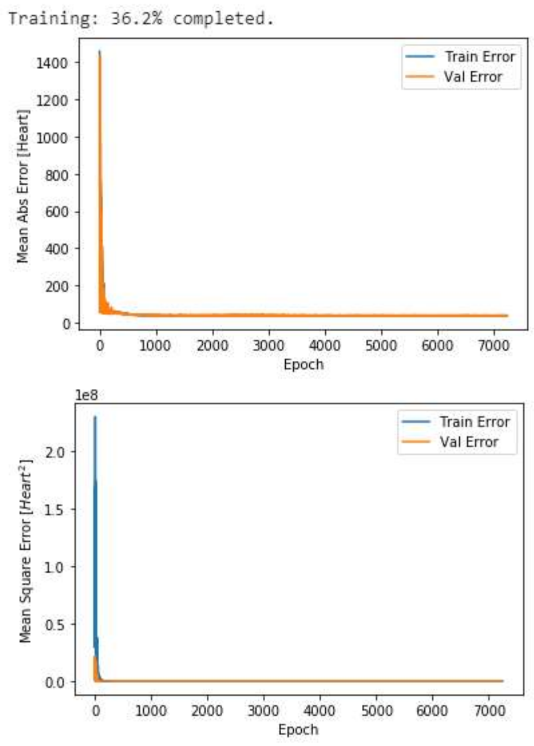
For the experiment, 20% of the dataset was taken to be used as test set, so the 80% remaining will be used as training, and then tested against the 20%. The selection is random, and the following lines of codes represent the actions taken over the dataset to split the data and then confirm it. In total 3,517 samples will be used as training dataset and the rest as test dataset.



```
train_dataset = dataset.sample(frac=0.80,random_state=0)
test_dataset = dataset.drop(train_dataset.index)

##we print a simple example to see how the data changed
sns.pairplot(train_dataset[["BMI", "Age","Heart", "Sodium@Referral"]], diag_kind="kde")
#"Liver", "R Kidney", "L Kidney"

# now we inspect and create the overall statistics
train_stats = train_dataset.describe()

#removing heart from training set.
train_stats.pop('Heart')
train_stats = train_stats.transpose()
train_stats
```

Figure 12 Section of code where we split the data and then confirm the samples.

At this point the normalization method can be selected, L1 or L2 or neither. At first attempts was decided to not normalize the data and observe how the model would behave.



```
Training: 36.2% completed.
```

Figure 13 Result after training.

V. Results for heart recovery forecasting

Approximately 879 cases were left for testing, these cases have never been seen by the model during the training, but they were used as test for forecasting, passing by the model getting a forecast then compared to the real result.
Using the 879 cases as a test then calculate the difference between the forecasted number vs the real number, when successful recovery of the organ was performed the real number the model was looking for was a 100; when the recovery was not successful the number to be forecasted corrected was 0. Then to subtract each case, if the model forecasted a 20 and the expected result was a 100, then there was a margin error of 80. If the forecasted result was 1 and the expected result was 0, there was a margin error of 1, and so on. The following figure displays an example of the distribution of the margins of errors, where most of the errors were close to 0, but in an optimal result would be that all of it would be piled up in 0, indicating that all of them or most of them have a margin error of 0.

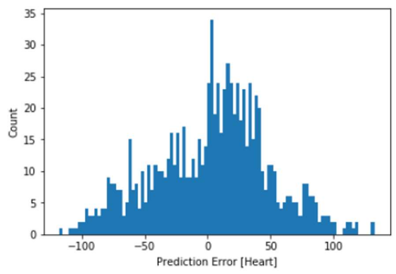
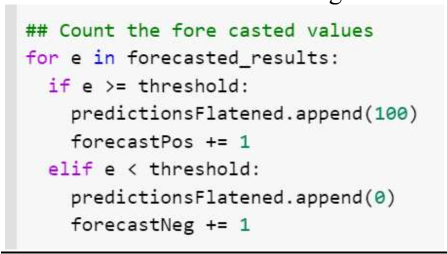


Figure 14 Margin error distribution.

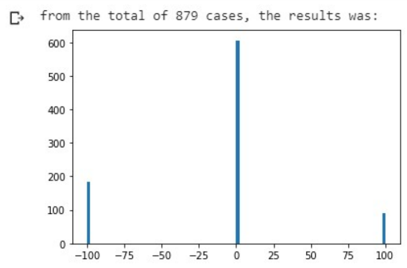
Since the model is returning a number between 0 and 100, instead of 0 or 1 or 0 or 100, it was decided that a threshold may be useful to flat the results into either positive or negative. Using the ROC and AUC calculations the threshold was set for now at ***55***, refer to [2] and [3].
The code to flat the values using the threshold would be as follows.



```
## Count the fore casted values
for e in forecasted_results:
  if e >= threshold:
    predictionsFlatened.append(100)
    forecastPos += 1
  elif e < threshold:
    predictionsFlatened.append(0)
    forecastNeg += 1
```

Figure 15 Code where threshold is set to flat the results, then also a confusion matrix is built to gain more insight on the results.

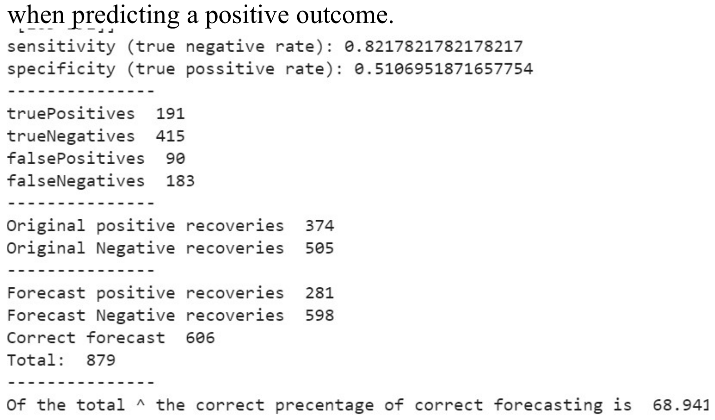
The distribution after the threshold set to 55 is used is as follows:



```
from the total of 879 cases, the results was:
```

Figure 16 Distribution chart of the differences between the forecast and the real result. Most of the values display a 0 difference that indicates most of them were correctly forecasted.

The figure displays that most of the results where correctly forecasted as they are piled over the 0 margin of error. The following confusion matrix helps us see more of the details about the accuracy of the forecasting of the model.
The confusion matrix helps determine the true negative rate was high, indicating that the model was good when forecasting when a recovery was not going to be successful. But it also indicates that when the model said it was not going to be a successful recovery, 82% of the time it was right, but only a 51% of the time was right when predicting a positive outcome.



```
sensitivity (true negative rate): 0.8217821782178217
specificity (true possitive rate): 0.5106951871657754
---------------
truePositives  191
trueNegatives  415
falsePositives  90
falseNegatives  183
---------------
Original positive recoveries  374
Original Negative recoveries  505
---------------
Forecast positive recoveries  281
Forecast Negative recoveries  598
Correct forecast  606
Total:  879
---------------
Of the total ^ the correct precentage of correct forecasting is  68.941
```

Figure 17 Screenshot of the results calculating the specificity and sensitivity of the results.

	True	*False*	*Total*
Negative	415	183	598
Positive	191	90	281

Figure 18 Confusion matrix with the result of the forecast for heart outcomes.

Reading the numbers, you can state the following:

I. Out of the 879 cases used in the test, 374 were actual positive recoveries, and 505 were not positive recoveries.
II. The model forecasted accurately 82.17% of the negative cases. Since 415 times of the 505 cases the model said it was going to be an unsuccessful recovery.
III. The model barely performed better than a coin toss in predicting positive outcomes with a 51.06% of times getting the right forecast.
IV. Out of the 879 cases, the model predicted correctly the outcome of 599 cases, for a 68.94% accuracy overall.
V. Out of the 879 cases, 505 were not positive recoveries, making it close to a 74% of the cases.

Exploring other thresholds
The following table shows the difference between several thresholds to be considered.

Threshold	*Overall%*	*TNR%*	*TPR%*
35	67.80	58.01	81.01
40	68.71	64.15	74.86
45	69.62	72.27	66.04
50	**70.07**	78.41	58.82
55	67.80	82.37	48.12
60	66.09	87.32	37.43

The conclusion for heart outcomes: the model was at best at a 70.07% overall success rate when forecasting successful recoveries.

VI. Results for Liver recovery forecasting

When forecasting liver successful recovery, it seems the model stopped faster and was unable to find a pattern for it.
After removing the null values (rows where liver recovery was null) instead of the same 3,517 it got 3,498 cases for training and 875 cases to test. The results are as follows:
From the 20,000 epochs requested to be run, it found the best weights and values after 2,000, continued running until approximately 7,000 and reverting to the best values found.

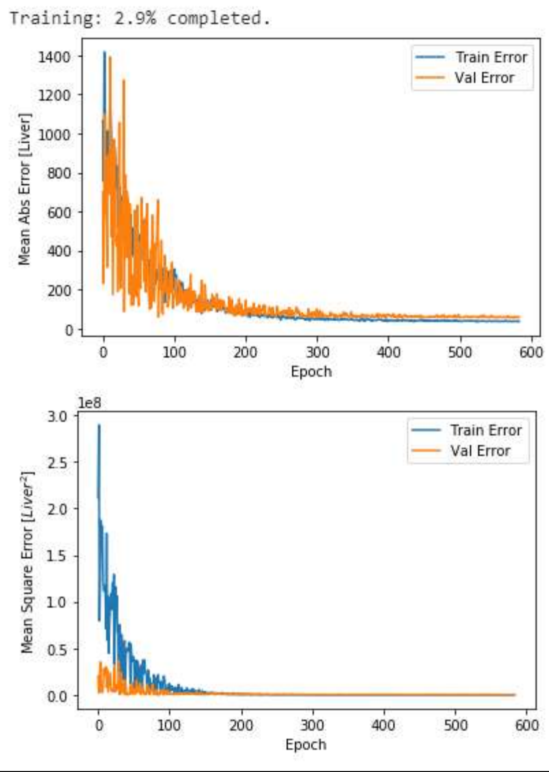

```
Training: 2.9% completed.
```

Figure 19 Entropy chart when learning.

The flattened results are displayed in the next figure where it took as a reference a threshold of 5 and it says that 75.77% cases where correctly forecasted, but it is also worth notice that 76% of the results are positive recoveries.

	True	*False*	*Total*
Negative	15	189	204
Positive	648	23	671

Figure 20 Confusion matrix with the result of liver outcomes forecasting.

Based on the confusing matrix, the model had a 96.75% success when forecasting a positive recovery, and 7.35% when forecasting a not successful recovery. (It said yes to everything)

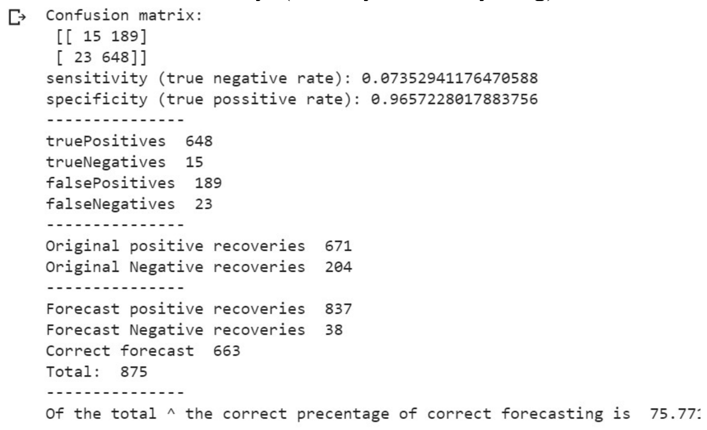

```
Confusion matrix:
 [[ 15 189]
 [ 23 648]]
sensitivity (true negative rate): 0.07352941176470588
specificity (true possitive rate): 0.9657228017883756
---------------
truePositives  648
trueNegatives  15
falsePositives  189
falseNegatives  23
---------------
Original positive recoveries  671
Original Negative recoveries  204
---------------
Forecast positive recoveries  837
Forecast Negative recoveries  38
Correct forecast  663
Total:  875
---------------
Of the total ^ the correct precentage of correct forecasting is  75.77
```

Figure 21 Screenshot of the result using 5 as a threshold for Liver.

Threshold	*Overall%*	*TNR%*	*TPR%*
1	75.2	5.39	96.42
2	**75.31**	5.88	96.42
10	74.97	7.35	95.52
30	72.22	12.74	89.71
35	70.97	12.74	88.67
65	59.31	39.70	65.27

In conclusion, for Liver outcomes, the model is unable to forecast the recovery, since it learned to say yes to everything, and the success rate match the overall forecasting rate, and the true negative is low, the model is not useful.

VII. Results for right kidney recovery forecasting

The experiment was changed to have a patience test of 500 and 2000 epoch to reduce the time of training, this change the setup to last less than 90 minutes from what it used to be approximately 4 hours.
After removing the null values for rows and columns the number of cases for training were 3,403 cases with 851 cases for testing.
The results were better than the other experiments, in this case the mean average error went down to 27, which made this the most successful experiment so far.

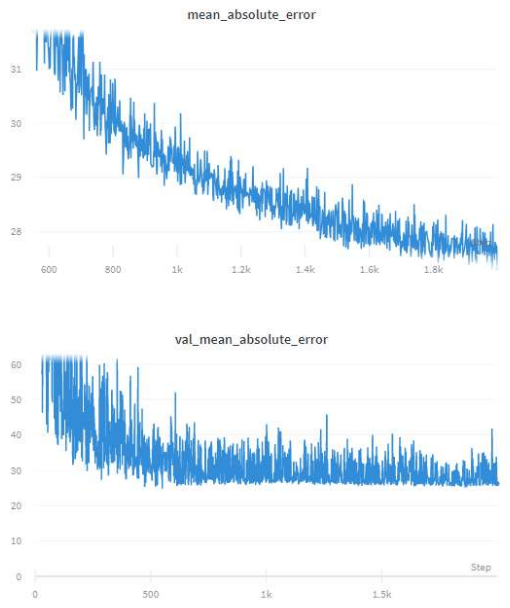


Figure 22 Mean absolute error of the forecasting during the training.

The following table shows multiple thresholds and where it could help determine the forecasting skills of the model. Where TNR is true negative rate and TPR is true positive rate.

Threshold	*Overall%*	*TNR%*	*TPR%*
50	79	13.36	99.38
55	79.31	16.33	98.92
60	80.14	20.29	98.76
70	**82.02**	37.12	95.99
72	81.66	41.58	94.14
75	80.14	49.50	89.67
80	74.14	58.91	78.89
82	72.38	64.35	74.88
85	68.03	70.29	67.33

Depending of the goal of the forecaster, the threshold may be set to either 70 or 85. 70 proved to have the best average overall forecasting ratio but with a low true negative ratio.

In conclusion, the model proved to have a good ratio for forecasting right kidneys compared to the previous experiments.

VIII. Results for left kidney recovery forecasting

After clearing null columns and rows the total examples for training were 3,404 (one more than the examples used for right kidney forecasting). After one hour of training and 2000 epochs the results of the multiple confusion matrixes are as follows:

Threshold	*Overall%*	*TNR%*	*TPR%*
50	80.61	16.84	98.94
55	82.25	25.26	98.63
60	83.19	36.31	96.67
61	**83.54**	40	96.06
62	82.84	42.64	94.40
70	72.38	62.63	75.18
72	69.80	68.94	70.04

Using a threshold of 61, the average successful forecasting is 83.54% but with a low TNR, but when using 72 as threshold the success rate goes down but the NTR goes up significantly to be 68.94% and a TPR of 70.04%.

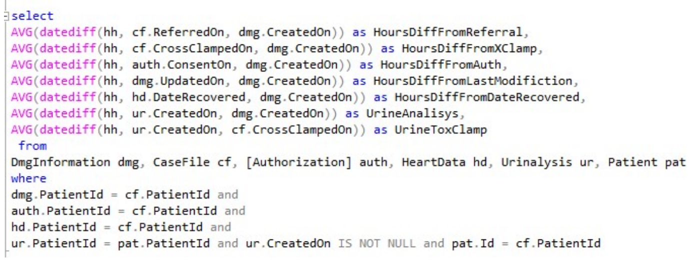
Conclusion: depending of the threshold you pick either 72 or 61, it can yield a significant ratio of correct forecasting, the model learned to find patterns on the left kidney recovery. It is still pending to see with more data or more time training.

IX. Analyses of the Dataset and Results

The true negative and the ratio overall seems to be good, maybe not for medicine, but in machine learning terms it found a pattern and it sems to be worthwhile to explore and see if a different shape of model, maybe different or more data or a different activation function, would be able to learn and forecast.
On this section, the experiment is looking to analyze any data used on the model and determine if that data is providing hard or strict bias and it is not an actual prediction and it is actually using simple flags hidden in the data to just say is or not a successful heart recovery.
For this it will be analyzed:

I. Is the data being fed to the model entered after the recovery was performed.
II. Audit records related to data being changed after cross clamp.
III. Install weight and biases (wandb.com) into the project to gain a better perspective of the model.

DmgInformation
The DmgInformation table contains basic information that was not actually used when training the model. So extensive research on the creation or modification may not be relevant.



```sql
select
AVG(datediff(hh, cf.ReferredOn, dmg.CreatedOn)) as HoursDiffFromReferral,
AVG(datediff(hh, cf.CrossClampedOn, dmg.CreatedOn)) as HoursDiffFromXClamp,
AVG(datediff(hh, auth.ConsentOn, dmg.CreatedOn)) as HoursDiffFromAuth,
AVG(datediff(hh, dmg.UpdatedOn, dmg.CreatedOn)) as HoursDiffFromLastModifiction,
AVG(datediff(hh, hd.DateRecovered, dmg.CreatedOn)) as HoursDiffFromDateRecovered,
AVG(datediff(hh, ur.CreatedOn, dmg.CreatedOn)) as UrineAnalisys,
AVG(datediff(hh, ur.CreatedOn, cf.CrossClampedOn)) as UrineToxClamp
 from
DmgInformation dmg, CaseFile cf, [Authorization] auth, HeartData hd, Urinalysis ur, Patient pat
where
dmg.PatientId = cf.PatientId and
auth.PatientId = cf.PatientId and
hd.PatientId = cf.PatientId and
ur.PatientId = pat.PatientId and ur.CreatedOn IS NOT NULL and pat.Id = cf.PatientId
```

The results for this are shown in the next table where it indicates the number of hours passed from the event listed to the creation of the DMG

	Referral	*xClamp*	*Authorization*	*Last DMG Modification*	*Heart Recovered*
lopa	572	496	536	-8	548
tosa	107	10	51	-142	20
ctdn	292	187	258	-490	133

- DMG record is irrelevant here since the data that is at the end in the model does not come from DmgInformation table.
- DMG records are created several hours/days after the cross clamp was performed.
- In average cross clamp is performed 78 hours after referral.
- In average the **urine analysis** data starts to be collected 35 hours before the cross clamp, which makes it viable data for forecasting.
- **ABG** data starts to be collected in average 33 hour before the cross clamp.
- **Angiography** data starts to be collected in average 3 before the cross clamp but is important to notice that some OPOs enter this information days after.
- **Hemodilution** information starts to be collected in average 39 hours before cross clamp.
- **Chemistry** data starts to be entered 44 hours before cross clamp.

In average the DMG is created several days after the cross clamp was performed, but the urine analysis data is collected about 34 hours before the cross clamp in average
So there is a lot going on before the DMG is created, using strictly the DMG for forecasting seems not to be viable, but still, during the experiments, the information used was not obtained from the DMG record, going back to the figure 8, none of the columns values are actually entered when DMG is created, they are supposed to be existing by then.
As a recommendation, ignoring the DMG record when collecting data would be a better option, this way it is possible to gather even more samples.

Data used as inputs
Figure 8 displays the data used as input for the forecasting. Reviewing the audit logs, the data was entered before the cross clamps and did not receive significant updates after the recovery. Most of the data comes from urine analysis, arterial blood gases, angiography and hemodilution.

Data Name	Type
Admission Date Time	Int, Minutes since referral
Authorization Date Time	Int, Minutes since referral
Height	int
Weight	int
BMI	number
Age	int
Sodium at Time of referral	number
Glucose at time of referral	number
Dose at time of referral	number
Dose at time the OPO starts	number
Dopamine at referral	number
Dopamine at OPO starts	number
Neosynepryne at referral	number
Neosynephrine at OPO starts	number
Levophed at Referral	number
Levophed at OPO starts	number
Epinephrine at Referral time	number
Epinephrine at OPO starts	number
Dobutamine at time of Referral	number
Dobutamine at time of Referral	number
Creatinine at referral	number
Levothyroxine at referral	number
Levothyroxyne at OPO starts	number
Vassopressin at Referral	number
Vasopressine at OPO starts	number
Latest Insuline at referral	number
Latest Insuline at OPO	number
Insuline at referral	number
Insuline at OPO starts	number

X. CONCLUSIONS

- The model was unable to forecast liver recovery, but, more examples and data from outside the DMG framework may improve the results.
- The model was successful when forecasting other organ outcomes, but it could still use more training and modification on some of the variables set up. Still with this result it may be helpful for resources management in OPOs.
- Going outside the DMG framework and collecting data from other database fields of the system may help improve the model, when this data is collected before the recovery happens, this will help collect a significant number of cases for the model.
- May help to ignore records where the data was entered after the cross clamp.

XI. REFERENCES

[1] Towards data science. https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c

[2] Statistical Thinking, *Machinelearning for Predictions for Healthcare The Confusion Matrix is a matrix of confusion*. https://www.fharrell.com/post/mlconfusion/ .

[3] Understanding AUC and ROC. Towards Data Sicence. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

Appendix 1

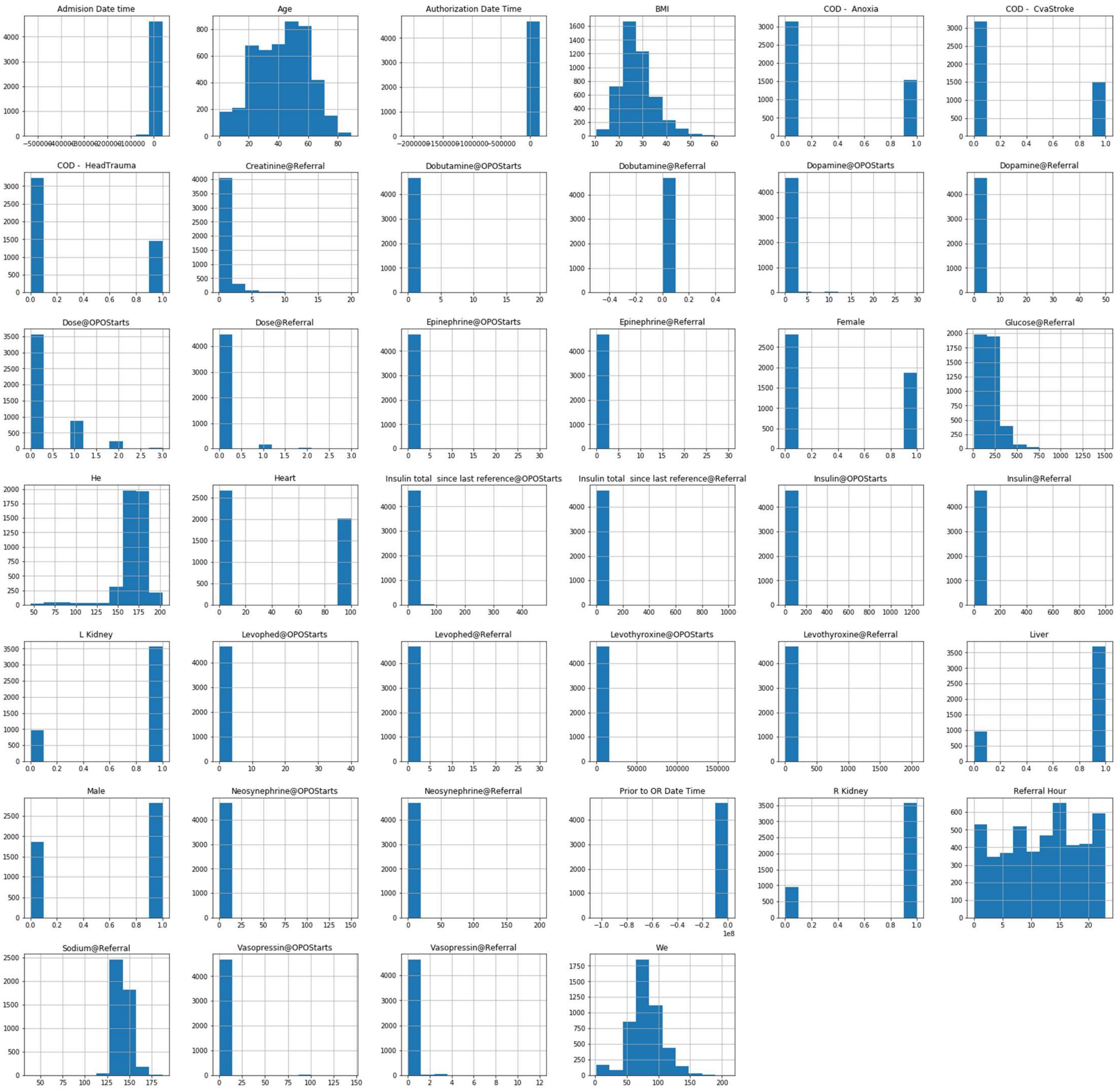


Figure 23 Distribution of the datapoints.

Appendix 2

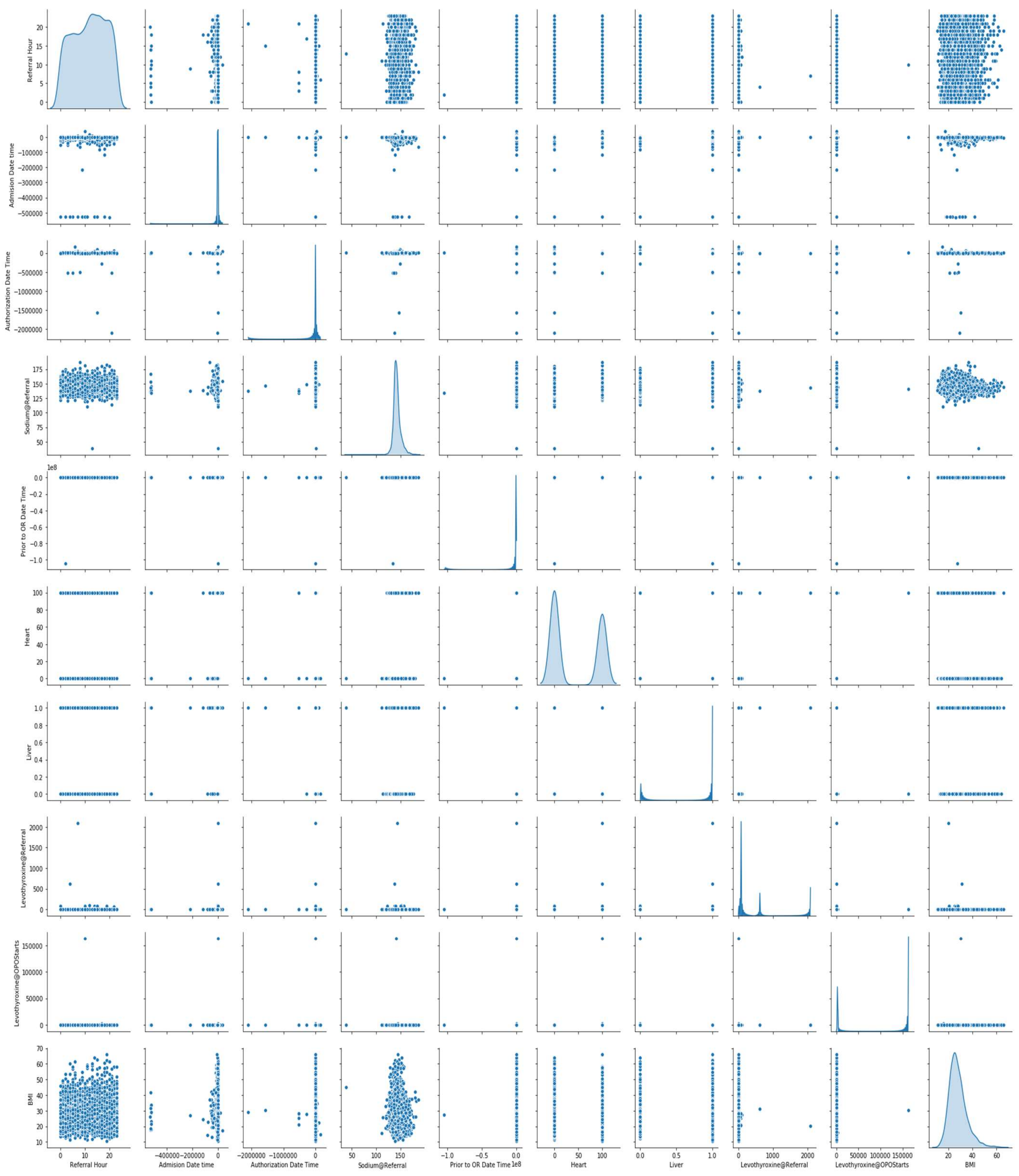


Figure 24 Pairplots of the datapoints.