Availability of downstream beds: downstream services for emergencies.

Pablo Aldana[1]

[1] *Centrale Lille, Lille, France*

February 28, 2021

Abstract

I studied how can we predict the availability of downstream beds in a hospital. For this I research 3 solutions pointing to a different perspective each. First is doing a real-time indoor tracking patient with the help of beacon's technology. The second is to predict the time of stay of a patient with a Logistic Regression model, to know when is he leaving the Department. Finally is to predict a specific state of a bed with Reinforcement Learning using Q-Learning. With the target to propose a model with this 3 elements to make a precise prediction of the availability of beds.

1 Packages Used

The code is avaible in Github

- Python 3.8
- Sklearn
- Pandas
- Tensor Flow Keras

2 Introduction

In the healthcare system, there are a lot of operations that demand complex interactions between different hospitals, departments, personal and patients. With the objective to provide a better service to patients and improve quality of healthcare we need to develop and adopt new technologies. For this we must ensure the correct collection and usage of electronic health record (EHR). The problem of overcrowding in the Emergency Department (ED) is one of majors problems affecting to the medical system recognized as an international problem (Hoot and Aronsky, 2008). To make more efficient the ED we need to improve the process involved.

One of process that involve ED and the others departments is the assignment of a bed for a new patient, today this process is done physically. Health care's personal need to consult by telephone or email if there is a bed available in the desired Department, making loose time, resources and more important a bed in ED used by the patient that need to be boarded. Because of this, various study focuses on how to improve this process and making it quicker. To cite one, (Lee et al., 2021) proposes a proactive in patient decision for boarding from ED to another department, using EHR for taking the decision. But for this type of implementation, we need to know for advanced where there is a bed available.

To predict, or know at the advance where is a bed available or when will be we can have three possible solutions. The first one is to do an indoor tracking of a patient for knowing when the patient leaves the bed

and make it available. The second one is to predict at the advance the time of stay (ToS) of a patient when is arrived using EHR. And finally the third one is to predict the state of a bed using Reinforcement Learning where we present the environment at the hospital and there is an agent who will take the decision of each state for each bed.

2.1 Indoor Tracking

For the Indoor Tracking there are two articles interesting to analyze who realized a deploy in a hospital environment to track a patient in real time. In the two studies they deploy beacons in rooms, doors and hallways. Also, for each patient they installed an app or put a bracelet that would receive signals from beacons for the tracking system. The difference between the two studies is how they predict the placement of the patient.

For the first one (Calderoni et al., 2015) they classified the rooms in clusters to makes zones, as we can see in Figure 18. For then when received signals from the app of the patient, they predict for each cluster which room he can be, and the one with more probability it would be. The result where a high accuracy, but the problem that is the environmental scale, so the number of grouping also increase, the time of prediction is high.
For the second study (Wyffels et al., 2014) they receive the signals of each patient an do an use a decision tree for search where the patient is, we can see in Figure 2. For this solution, it's important to deploy beacons in rooms and corridors for a more accurate prediction, so it demands more quantity of beacons.

2.2 Time of Stay

To predict the time of stay it's important to have access to EHR for each patient, with the detail of his medical record, severity of illness etc... There are different study that uses different approach to achieving a good prediction. For example (Cai et al., 2015) studies how accuracy improves when more data on the patient is obtained, using a Bayesian Network. In another example of study (Gentimis et al., 2017) predict the ToS using a Neural Network, with data from MIMIC III for knowing if the patient will be ≤ 5 days or more. Also, there is a review of different algorithms and their accuracy in the ToS prediction (Fernandes et al., 2020) where they conclude that an approach with Logistic Regression can achieve a score of 0.80 in Average AUC.

2.3 Reinforcement Learning

For the approach with the reinforcement Learning, the model learns by himself when the environment gives him a reward or a penalty for taking some decisions.

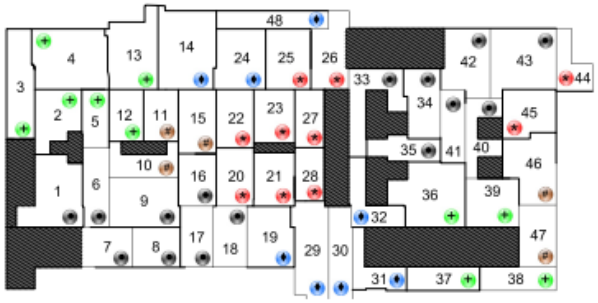

Figure 1: *Example of room grouping. Calderoni et al., 2015*

For example, in medical triage there is a study (Buchard et al., 2020) that train an agent for predicting the triage for patients in the ED. The agent can classify each patient by a color system that indicate the severity of his illness. The problem with this type of algorithms is the time of learning that takes more time compared to Logistic Regression for example. But it can be done with less amount of data, because is the agent that learn about the environment and himself generate the data.

3 Data set

For the study I take a Data set available in Kagle: Health Care Analytics - 2. The data come from the Janatahack contest, where the objective is to predict the ToS of a patient. In the data we use the train.csv for training and then test.csv for the different application proposed. In the data there are 18 columns, but for the learning I used only 11:

1. Available Extra Rooms in Hospital
2. Bed grade
3. Visitors with patient
4. Number of times in the past
5. Ward type
6. Ward Facility Code
7. Department
8. Severity of Illness
9. Type of Admission
10. Age
11. Stay

The column "Number of times in the past" that try to simulate an historical record of the patient was added by me. For this I ordered each case of patients by their case id, and assuming that the lower the case id was, it was more in the past. So this way I count for each patient the time he goes to the hospital.

Also in the database some columns where categorical, for the machine learning purpose I converted into integer with a correspondence for each category using the library pandas from python. In the column of stay, there are 11 different categories from "[0,10]" days to "More than 100 days". So the prediction of stay will not

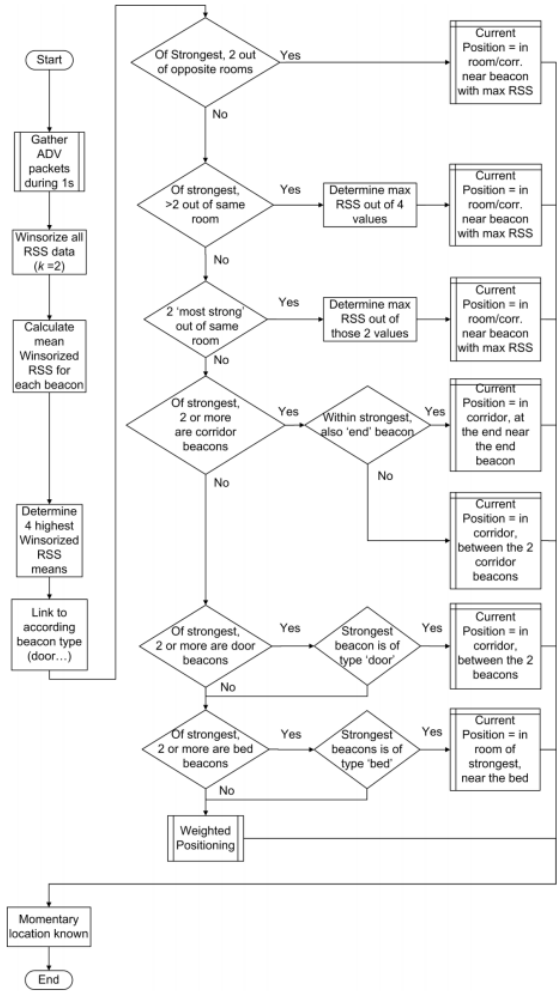

Figure 2: *Decision Tree for Indoor Tracking Wyffels et al., 2014*

Algorithm 1 Get historical record of patient

Require: data
  data ← data sorted by patient id and case id
  $n \leftarrow$ initialize an array with 0 with len of the data
  for 1 ... len of data **do**
    pid ← id of patient i in data
    if pid = id of patient $i-1$ in data **then**
      $n[i] \leftarrow n[i-1]+1$
    end if
  end for
  data ← We append n as a new column

Table 1: *Statistical description*

Column	Mean	Sd	Min	Max
Available Extra Rooms	3.19	1.17	0	32
Bed Grade	2.63	0.87	1	4
Visitors with Patient	3.29	1.76	0	32
Number of time in the past	2.03	2.27	0	49
Severity of Illness	1.37	0.77	0	2

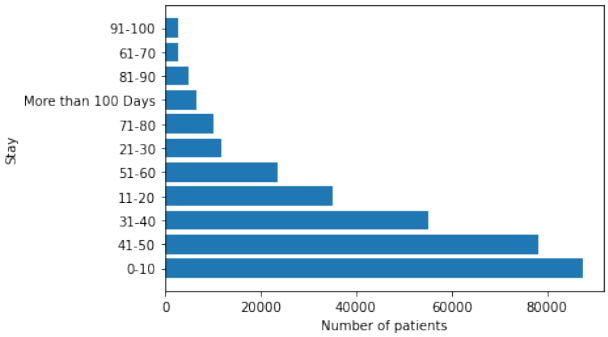

Figure 3: *Frequency of stay*

be precise in a specific day, but in a group of days when the patient is admitted. Overall summary statistics are provided in Table 1.

We can see in Figure 3, 4, 5, 6, more detail of the data. Particularly in Figure 5 we saw a difference between Urgent and Emergency this due the USA health care system that make a difference in the severity of illness of a patient. In Emergency the case needs a more intensive and quicker attention compared to Urgent. Also, we saw that in Figure 6 there is a dominance in patients that are in radiotherapy, this can cause a bias for the machine learning.

4 Procedure

4.1 Time of Stay

To build the model that will predict the ToS, I tried with three different algorithms continuing with the articles

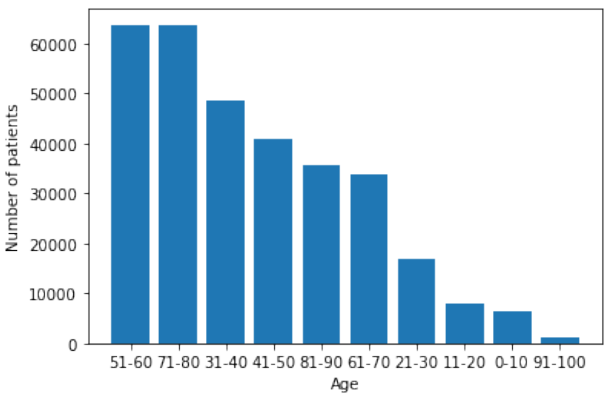

Figure 4: *Frequency of Age*

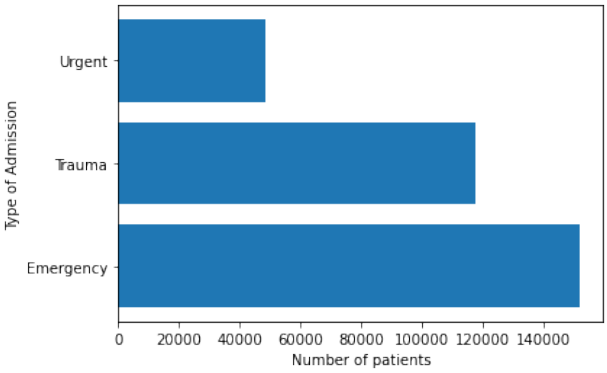

Figure 5: *Frequency of Type of Admission*

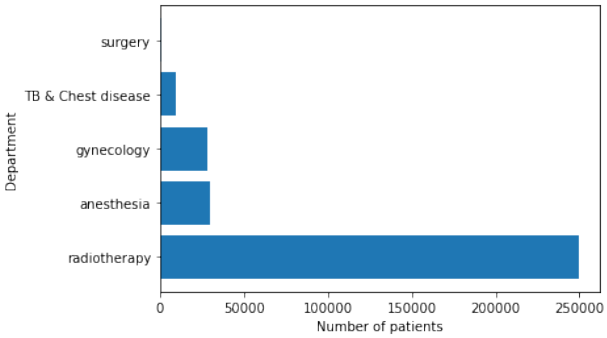

Figure 6: *Frequency of Department*

already mentioned. First, I train a RandomForest with the library Sklearn from python. In second I built a Neural Network (NN) with the same proprieties of (Gentimis et al., 2017). This NN contain 3 layers, the first one with 5 neurons the second with 3 and the last with a softmax function activation. I used as optimizer the Adam function and for the loss the Categorical Cross Entropy. It was built in python with the library TensorFlow Keras. Finally, I implemented a Logistic Regression with Sklearn in python.

For choosing the best of the three models I compare the ROC curve for each class and the AUC score. In the Table 2, we see the mean score for each model. Because logistic obtained a better result, it will be implemented in the application. In the Appendix is available all the ROC curve in detail for each class.

A direct application is to see the availability of beds in the next few days for a specific hospital. For this I take the data from test.csv and take some patient from a same hospital, simulating that they are all at the same time. So now with the model we can know how

Table 2: *AUC mean score for each model*

Model	AUC score
Logistic Regression	74%
Neural Network	67%
Random Forest	71%

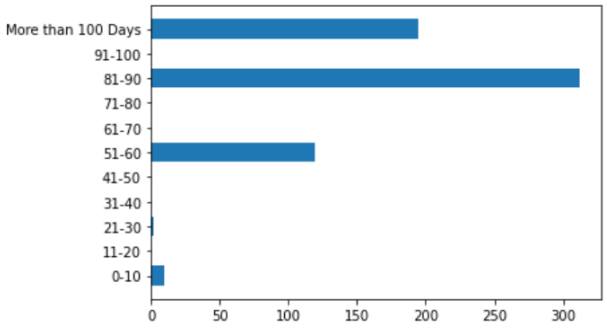

Figure 7: *Prediction of Stay for patients in Hospital 1*

the capacity will change in the next days. With these we can obtain for example for the hospital number one the Figure 7. Also, we could easily predict for just a patient his ToS just when he arrived to the hospital, or also see the comportment in the different Department from a same Hospital.

Algorithm 2 Predict the availability of bed for an hospital i

Require: data, Logistic Regression model trained
data ← data from the hospital i
$prediction$ ← model(data)
plot(prediction)

4.2 Reinforcement Learning

We implement reinforcement learning using a Q-learning model. Where an agent will decide the different state for each bed. There are 4 possible states for a bed:

1. Occupied
2. Available
3. Reserved
4. In transition

The transition state is when just a patient leaves the bed, and personal of healthcare need to clean and tidy up for the next patient.

We define the environment as a hospital that have a defined number of beds and receive patient by the time. The agent will operate in the environment and take decision for each beds. If the decision is good he received a reward. To make it simple, each day we have a probability p that a patient arrives at the hospital, this one have a bed reserved and after one day he will arrive.

For each decisive step taken by the agent, the environment need to check if the decision was good, if it is the case, then he gives a reward to the agent. If the decision is wrong the environment give a negative reward to the agent, and updates bed status according a decision taken.

Algorithm 3 Hospital environment arrive of a patient

```
hospital ← array with n bed
arrive ← random(0, 1)
if p ≤ arrive then
    we assign a bed available i to the patient
    hospital[i] = 3 the bed is reserved
end if
```

Algorithm 4 Reward Step

```
Require: decision, check function, update function
result ← check(decision)
if result == True then
    agent.reward + = REWARD
else
    agent.reward − = PENALTY
end if
update(decision)
```

For the agent model, we initialize the Q-table for each episode that we run this way he will learn in this Q-table by each step. For each step he predicts a state for each bed and then he gains or loos a reward. This prediction can make done randomly with a probability ϵ. At the end of the step, there is a training function that will improve the model. The agent has a memory where he saves each action taken. Then for the training he can access the memory to train with the historical data set.

One episode will concern x day, where the environment receives patient and each day the agent take decision for all beds. At the end of a day the model train with the memory for a better improve, see Algorithm 5.

Algorithm 5 Episode

```
Require: Environment, Agent
Agent ← initialize Q-table.
for 1 ... X-days do
    environment receive a patient
    if ϵ ≤ random(0,1) then
        decision ← agent take a random decision
    else
        decision ← agent predict the decision
    end if
    environment check the decision
    agent.memory ← save the step
    agent ← train the agent
end for
```

5 The complete Model

The idea is to implement the three solutions for a complete Model in a hospital. This way we can verify if a bed is available with the different algorithms. The procedure is to check in real time the result of the three algorithms and if two have as result that a patient leave or his bed is available, then we conclude that the bed is available for a future use. This way we make a double check. The algorithm 6 is called for a double check when: Indoor Tracking don't find a patient making to think that the patient leaves the Department, ToS predicted that the actual day a patient end his stay at the Department or when the Reinforcement Learning change a state of bed.

Algorithm 6 Double Check for a patient

```
Require: Indoor Tracking, ToS, Q-learning, pid: id patient
result ← 0
if predict(Indoor Tracking for pid) = True then
    result ← +1
end if
if predict(ToS for pid) = True then
    result ← +1
end if
if predict(Q-learning for pid) = True then
    result ← +1
end if
if result ≥ 2 then
    The bed of patient pid is avaible.
end if
```

In case only one of the algorithm give a result that the patient leave, we can make a manual check with the health personnel.

6 Conclusion

This study helps to implement an application for having a real-time prediction of the availability of downstream beds. The objective is to expedite the ED and optimize human and economical resource.

In a future study it would be appropriate to implement the algorithms in a real hospital. With data from this hospital for a better adaptation. This way, the results obtained from each algorithm can be analyzed and adjusted to the environment.

Bibliography

Buchard, Albert et al. (Mar. 2020). "Learning medical triage from clinicians using Deep Q-Learning". In:

Cai, Xiongcai et al. (Sept. 2015). "Real-time prediction of mortality, readmission, and length of stay using electronic health record data". In: *Journal of the American Medical Informatics Association* 23.3, pp. 553–561. ISSN: 1067-5027. DOI: 10.1093/jamia/ocv110. eprint: https://academic.oup.com/jamia/article-pdf/23/3/553/34938618/ocv110.pdf. URL: https://doi.org/10.1093/jamia/ocv110.

Calderoni, Luca et al. (2015). "Indoor localization in a hospital environment using Random Forest classifiers". In: *Expert Systems with Applications* 42.1, pp. 125–134. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.07.042. URL: https://www.sciencedirect.com/science/article/pii/S0957417414004497.

Fernandes, Marta et al. (2020). "Clinical Decision Support Systems for Triage in the Emergency Department using Intelligent Systems: a Review". In: *Artificial Intelligence in Medicine* 102, p. 101762. DOI: 10.1016/j.artmed.2019.101762. URL: http://dx.doi.org/10.1016/J.ARTMED.2019.101762.

Gentimis, Thanos et al. (2017). "Predicting Hospital Length of Stay Using Neural Networks on MIMIC III Data". In: DOI: 10.1109/dasc-picom-datacom-cyberscitec.2017.191. URL: http://dx.doi.org/10.1109/DASC-PICom-DataCom-CyberSciTec.2017.191.

Hoot, Nathan R. and Dominik Aronsky (2008). "Systematic Review of Emergency Department Crowding: Causes, Effects, and Solutions". In: *Annals of Emergency Medicine* 52.2, 126–136.e1. ISSN: 0196-0644. DOI: https://doi.org/10.1016/j.annemergmed.2008.03.014. URL: https://www.sciencedirect.com/science/article/pii/S0196064408006069.

Lee, Seung-Yup et al. (2021). "Proactive coordination of inpatient bed management to reduce emergency department patient boarding". In: *International Journal of Production Economics* 231, p. 107842. ISSN: 0925-5273. DOI: https://doi.org/10.1016/j.ijpe.2020.107842. URL: https://www.sciencedirect.com/science/article/pii/S0925527320302085.

Wyffels, Jeroen et al. (May 2014). "Using a decision tree for real-time distributed indoor localization in healthcare environments". In: *2014 International Conference on Development and Application Systems, DAS 2014 - Conference Proceedings*, pp. 103–109. DOI: 10.1109/DAAS.2014.6842436.

A ROC curve for ToS algorithms

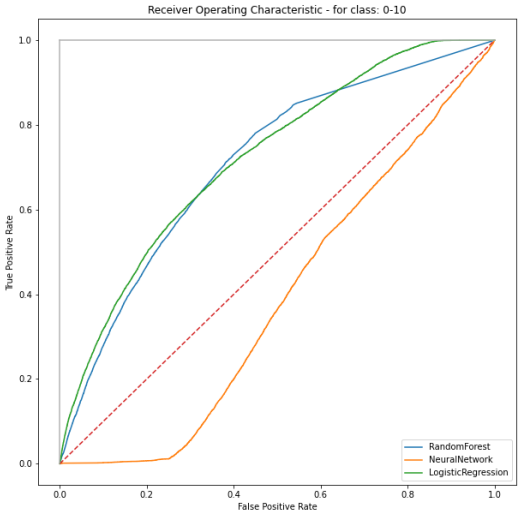

Figure 8: *ROC curve for class 0. Calderoni et al., 2015*

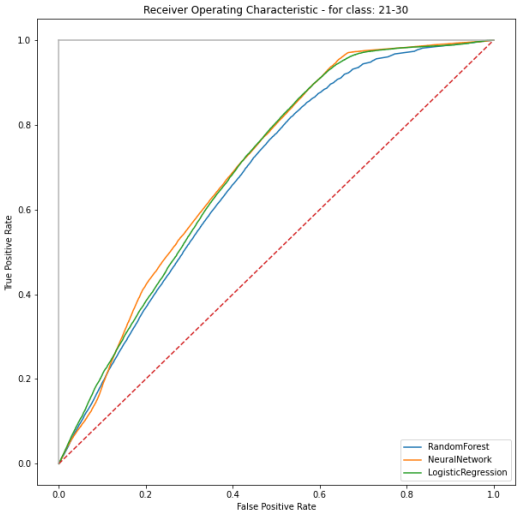

Figure 10: *ROC curve for class 2. Calderoni et al., 2015*

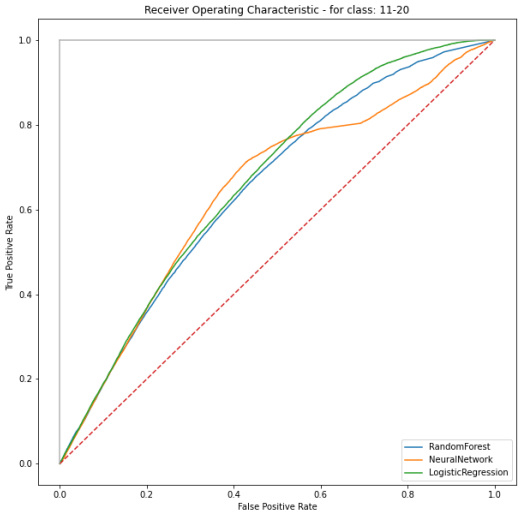

Figure 9: *ROC curve for class 1. Calderoni et al., 2015*

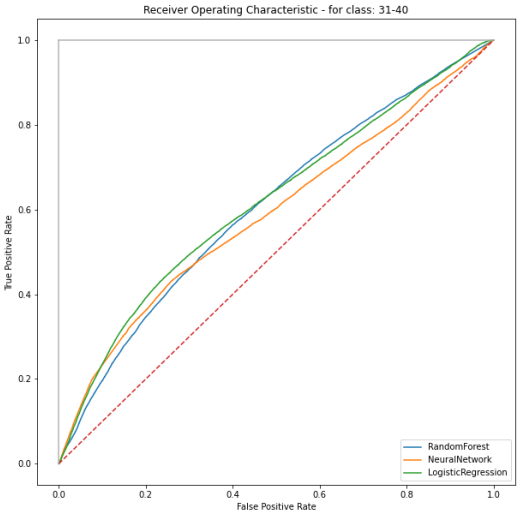

Figure 11: *ROC curve for class 3. Calderoni et al., 2015*

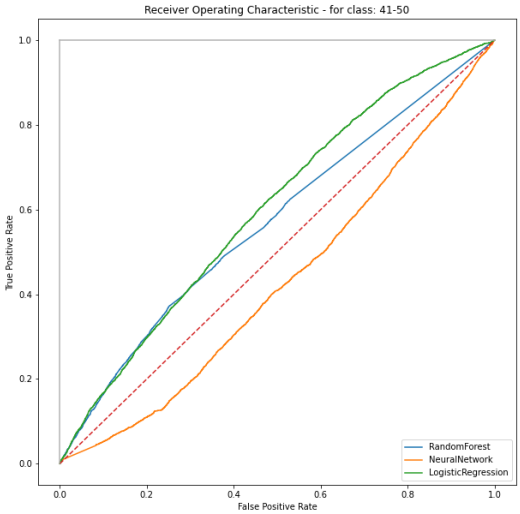

Figure 12: *ROC curve for class 4. Calderoni et al., 2015*

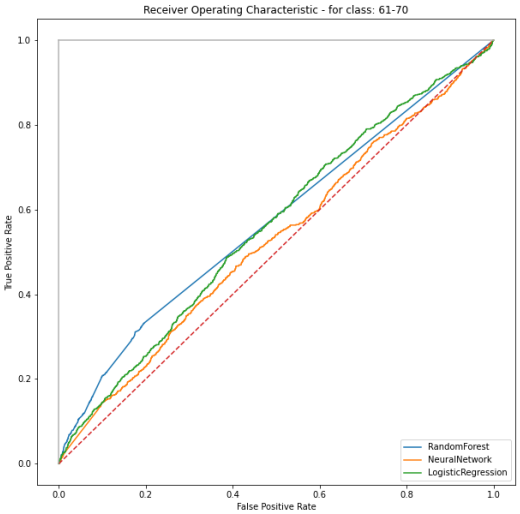

Figure 14: *ROC curve for class 6. Calderoni et al., 2015*

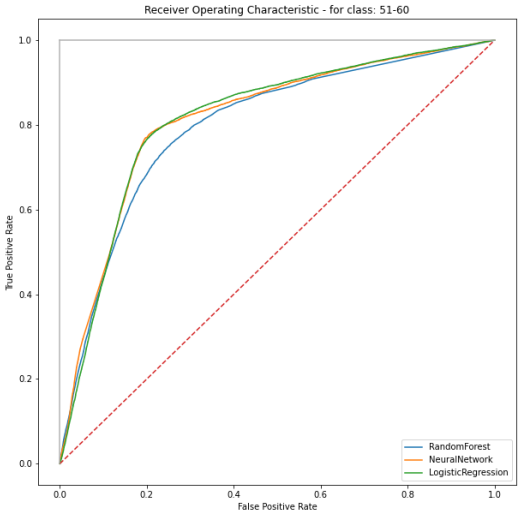

Figure 13: *ROC curve for class 5. Calderoni et al., 2015*

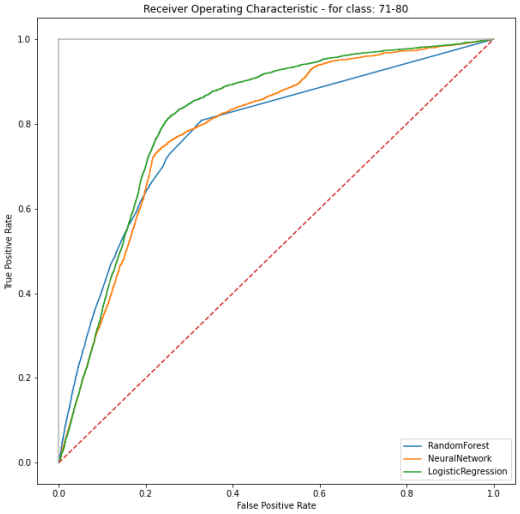

Figure 15: *ROC curve for class 7. Calderoni et al., 2015*

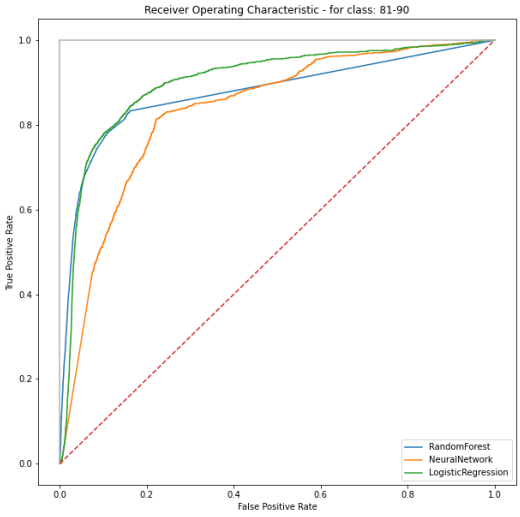

Figure 16: *ROC curve for class 8. Calderoni et al., 2015*

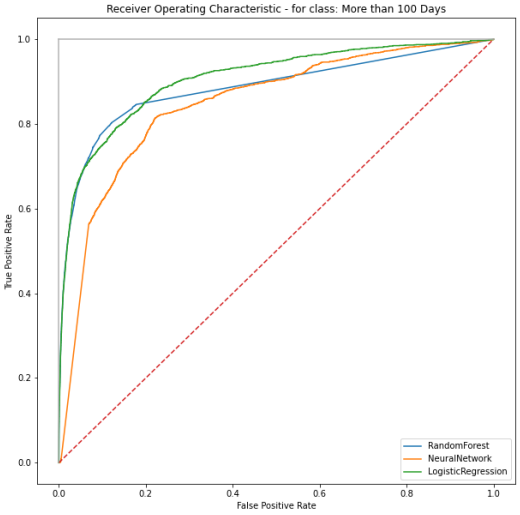

Figure 18: *ROC curve for class 10. Calderoni et al., 2015*

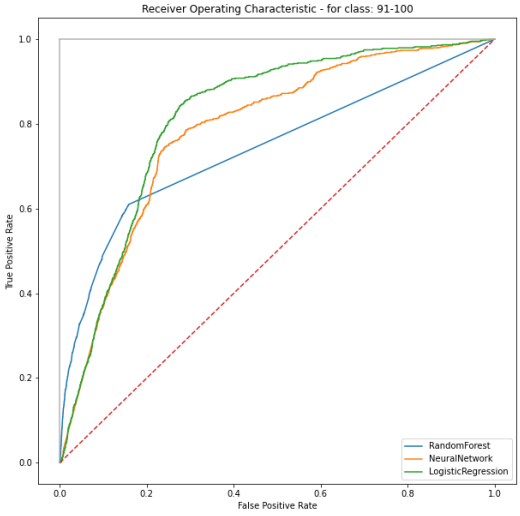

Figure 17: *ROC curve for class 9. Calderoni et al., 2015*