Hospital waiting times analysis
Streams Processing

Diogo Silva nº 53058 and Eugen Ursu nº 52533

Departamento de Informática
Faculdade de Ciências e Tecnologia
Universidade Nova de Lisboa

Introduction

Portugal has many public and private hospital which offer a vast range of specialties. These can include cardiology, gynecology, neurology, orthopedics, pediatrics and other more specialized that cover specific fields like oncology.

Overall, Portugal offers a good standard of medical care.

According to the National Statistics Institute (Instituto Nacional de Estatística), there are 238 hospitals in Portugal. More than half of these are private hospitals (127), and the rest are either public or partially funded by the state.

In this paper, we will focus in one of these hospitals, Hospital Santa Maria, located in Lisbon.

In a study, prepared by the Health Observatory of the European University, entitled "The Portuguese and Health in the post-pandemic", it is stated that 48.7% of respondents say they have difficulty accessing a specialist consultation and, among these, 30.2% have been waiting more than three months for one of these appointments[1].

As such, waiting times are the main reason people always seem to complain about and the main focus of our work.

We will use several data analysis tools and techniques in our dataset with the intent of visualizing our data and draw some conclusions from it.

State of art

Predicting waiting time to treatment for emergency department patients using more advanced techniques (in this case, machine learning algorithms) to improve waiting time forecasts instead of more traditional methods (rolling average or median estimators) which have less accuracy is the focus of the work studied. This is important as correctly predicting waiting times directly correlates with patient satisfaction and and smoother distribution of patient loads among health providers. Of course it could also backfire, as overestimation of waiting times can lead to patients leaving before receiving treatment they need.

As such, the dataset used for this was routinely collected administrative data from a single hospital, Princess Alexandra Hospital (PAH), for the period of 2 years (1 January 2016 to 31 December 2017). it contains information of the detailed time and clinical information about the patient's journey in ED (arrival → throughput → departure). The study was focused in the waiting time prediction for low acuity patients who have been assigned to the allocated to the waiting room and the other non-ambulatory waiting areas on arrival (3 to 5 in a 1 to 5 scale from urgent to low priority), as they accounted for 77% of all patients and high priority patients waiting times were clearly low, 1 minute for triage 1 patients and 10 minutes for triage 2 patients. The waiting time for the others, had large fluctuations in waiting time.

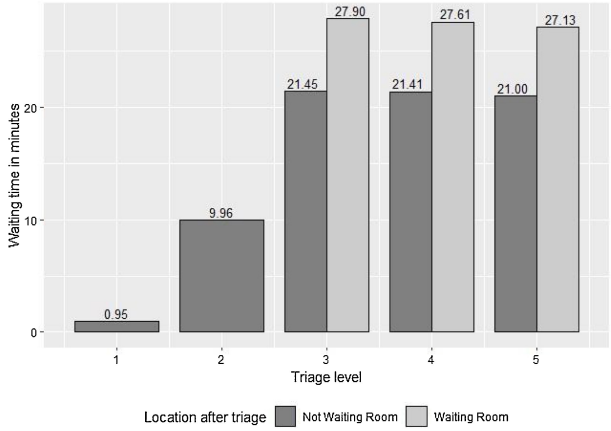

Figure 1: Average waiting time by triage and waiting room status

They concluded in this study that, "By implementing ML algorithms and using a large set of queueing and service flow variables, we provide evidence of the improvement in waiting time predictions for low acuity ED patients assigned to the waiting room"[2], the authors were able to ascertain that the ML algorithms outperformed even the best rolled averages by over 20% with respect to the mean squared prediction error (MSPE) and the quantile regression reduced the number of patients with large underpredicted waiting times by 42%, showing the advantages of using ML techniques which explore data connections in flexible ways, identify relevant predictors, and prevent overfitting of the data.

Dataset description

Title: Waiting times and people waiting in Lisbon Emergency Departments.

Description: Given the problem introduced, this is the perfect streaming data to analyse and process as it contains information about the waiting times and people waiting in some portuguese hospital emergency departments.

The data set contains 1048575 entities (with no missing data) and 8 features:

1. Acquisition time (Timestamp)

   It refers to the acquisition time of the data record.

2. Hospital (Categorical Nominal)

   It refers to a given Hospital.

   The values can take the form of: 211, 218, 216 and 208.

3. Urgency Type (Categorical Nominal)

   It refers to the type of the urgency department.

   The values can take the form of: Urgncia Polivalente, Urgncia Central, Urgncia Geral, Urgncia Peditrica.

4. Service (Categorical Nominal)

   It refers to the hospital services of the urgency departments

   The values can take the form of: cirurgia, pediatria, ortopedia, oftalmologia, etc.

5. Emergency stage (Categorical Ordinal)

   It refers to the severity of the emergency, 1 being the least severe and 4 being the most severe.

   The values can take the form of: 1, 2, 3 and 4.

6. Waiting time (Numeric)

   It refers to the amount of time (in minutes) in the last 2 hours that a person needs to wait to get medical attention.

7. People waiting (Numeric)

   It refers to the amount of people waiting on a service of an urgency department.

8. Hospital Name (Categorical Nominal)

   It refers to the name of a given hospital.

   The values can take the form of: Hospital Santa Maria, Hospital São José, Hospital Dona Estefânia and Hospital São Francisco Xavier.

Questions

A big question for us is:

Predicting waiting times in different urgency services and at different emergency stages at a given point in time in a hospital.

Methods Used

Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques, that does not include formal statistical modeling and inference. It is used to discover trends, patterns, check assumptions with the help of statistical summary and graphical representations, it is used to detect data noise, select data models, determine relationships between the explanatory variables and between explanatory and outcome variables.

We applied some techniques to analyze our dataset:

```
df.describe()
```

	Unnamed: 0	Hospital	Emergency_Stage	Waiting_Time	People_Waiting
count	2.648650e+05	264865.0	264865.000000	264865.000000	264865.000000
mean	5.163953e+05	218.0	2.598124	65.813124	3.904721
std	3.056496e+05	0.0	0.967595	70.611884	5.687532
min	8.000000e+00	218.0	1.000000	0.000000	0.000000
25%	2.477080e+05	218.0	2.000000	24.000000	0.000000
50%	5.136870e+05	218.0	3.000000	44.000000	2.000000
75%	7.817780e+05	218.0	3.000000	80.000000	5.000000
max	1.048596e+06	218.0	5.000000	599.000000	55.000000

With this we can for example say that the mean waiting time is 65 minutes and there are on average 5 people waiting.

To solve the research question we only took in consideration the waiting times in the hospital Santa Maria, and the services Cirurgia and Medicina, hence we removed some data from the dataset and removed the missing values, we also added a column 'lag_Waiting_Time_30_mn' with the waiting times of the last 30 minutes:

```
df2=df.loc[df['H_Name']=='Santa Maria']
df2.reset_index(drop=True, inplace=True)

Lag_value = 3 ## a data point every 10
    mn -> we look at the value on the
    last 30 mn (3*10 mn)
df3=df2[['Service','Emergency_Stage','
    Acquisition_Time','People_Waiting','
    Waiting_Time']].copy()
df3["lag_Waiting_Time_30_mn"] = 3

for em in range(1,6):
  df3.loc[(df3['Service']=="Cirurgia") &
       (df3['Emergency_Stage']==em),'
      lag_Waiting_Time_30_mn'] = df3[(df3
      ['Service']=="Cirurgia") & (df3['
      Emergency_Stage']==em) ].
      Waiting_Time.shift(Lag_value)
  df3.loc[(df3['Service']=="Medicina") &
       (df3['Emergency_Stage']==em),'
      lag_Waiting_Time_30_mn'] = df3[(df3
      ['Service']=="Medicina") & (df3['
      Emergency_Stage']==em) ].
      Waiting_Time.shift(Lag_value)
```

```
#new_df2['lag(Waiting_Time,30 mn)'] =
    new_df2[new_df2['Service']=="Medicina
    "].Waiting_Time.shift(3)
df3[['Service','Emergency_Stage','
    Waiting_Time','lag_Waiting_Time_30_mn
    ']].head(25)

df4 = df3.dropna().copy()

df4[df4['lag_Waiting_Time_30_mn'].isna()
    ], df4['Service'].unique()

services_to_keep = ['Cirurgia', '
    Medicina']
df5 = df4.query("Service in
    @services_to_keep")
df5[df5['Service']=='Aguarda Balcao
    Trauma']
```

	Emergency_Stage	People_Waiting	lag_Waiting_Time_30_mn
count	395723.000000	395723.000000	395723.000000
mean	2.601706	4.149817	69.618165
std	0.960108	6.292693	75.224115
min	1.000000	0.000000	0.000000
25%	2.000000	0.000000	24.000000
50%	3.000000	2.000000	45.000000
75%	3.000000	5.000000	85.000000
max	5.000000	55.000000	599.000000

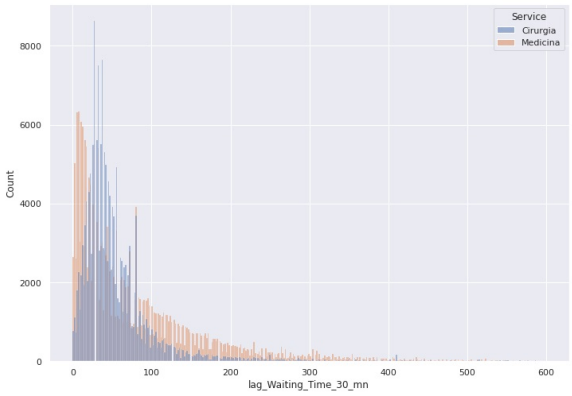

Figure 2: Waiting time of the last 30 minutes in Medicina and Cirurgia services

Concept Drift

Concept drift is a term used in predictive analytics and machine learning to describe how the statistical features of the target variable that the model is attempting to forecast vary over time in unexpected ways. This raises issues because as time goes, the forecasts grow less accurate.

To see changes over time in our dataset about the waiting time in the hospital, we used River's drift detectors to analyse our data. As with the state of art we divided the dataset into 2, low priority 1, 2, 3 on one side and high priority 4,5 on the other.

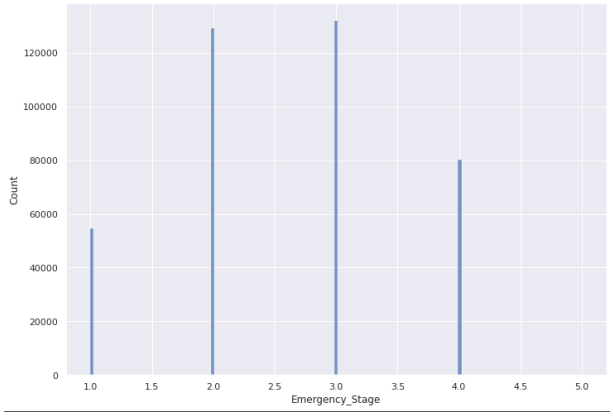

Figure 3: Emergency Stages Distribution

As we can see in Figure 3, the low high priority emergency stages 2 and 3 have similar numbers of patients admitted and the high priority emergency stage 5 however has a very low occurrence in the hospital (just 60 records) not showing in the figure nor giving us much information about that emergency stage.

We have tried several drift detector such as ADWIN, DDM and PageHinkley and obtained some interesting graphs:

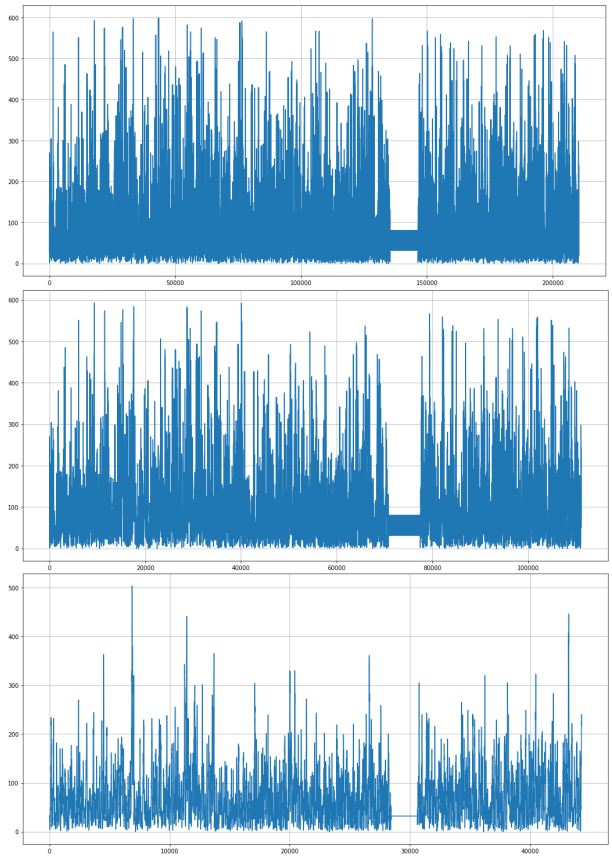

Figure 4: Emergency Stage 1,2,3; Emergency Stage 1,2,3 and Medicina; Emergency Stage 3 and Medicina (top to bottom)

The subsets Emergency Stage 1,2,3; Emergency Stage 1,2,3 and Medicina; Emergency Stage 3 and Medicina show almost the same waiting time increases over time. As such we decided to focus in one service: Medicina seeing as for the other, the process would be the same. As such for the drift detection we can use only the first one. To note however that subsets like Emergency Stage 1, will not show the same increases as the other 3 subsets, and also a time were there is not much variation (almost horizontal line at 3/4 of the graph) which we could not pinpoint why.

From River's drift detection methods, 2 are of note: ADWIN, studied in the classes, and PageHinkley. ADWIN is a more automatic method of detection with only the parameter delta (confidence value) while PageHinkley allows the user to restrict more parameters[6].

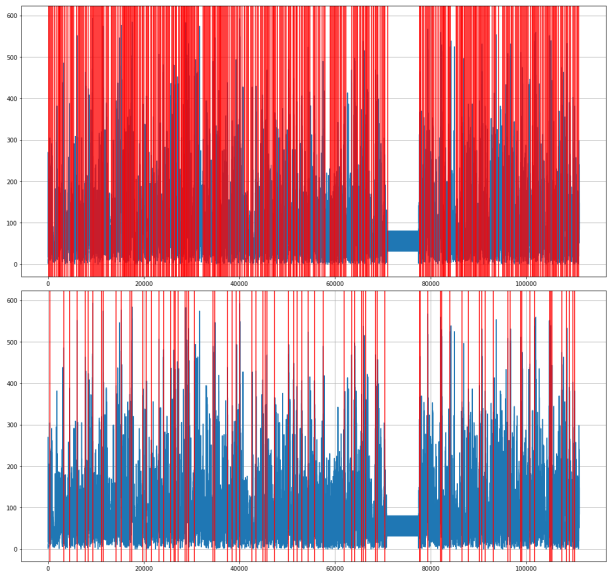

Figure 5: ADWIN vs PageHinkley for Low Priority

For ADWIN, even with delta = 1e-8, the method finds a lot of drift detection, and does not allow for a good visulalization. For PageHinkley, by fine-tuning it, we obtain much less drift detection, allowing us to visualize that most of it is due to large increases in waiting time.

For high priority 4 and 5 emergency stages:

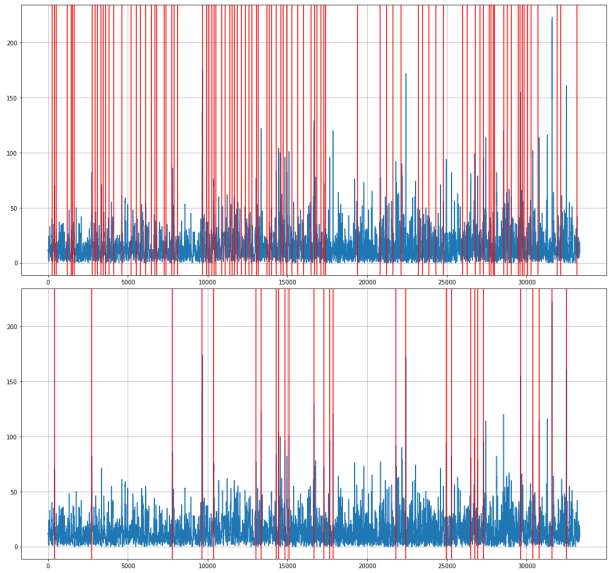

Figure 6: ADWIN vs PageHinkley for High Priority

We can also check the waiting times histograms between both priorities and both services:

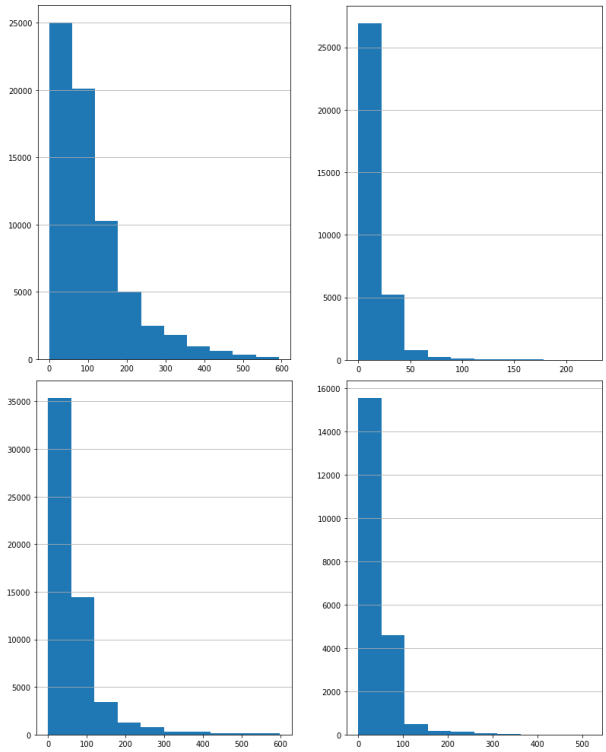

Figure 7: Waiting times histograms

Where in the top half we have low priority and high priority for the Medicina service, and in the bottom for the Cirurgia service, respectively. For low priority we have long waiting time, being longer in the Medicina service than the Cirurgia service while for high priority, the opposite happens.

Incremental and Online Learning

Incremental learning is a branch of machine learning that involves processing incoming data from a data stream—continuously and in real time, so we transformed our dataset into a stream with the help of stream.iter_csv documentation to apply an incremental model.

These algorithms are flexible, efficient, and adaptive and unlike traditional machine learning they can adapt, in real time, to changes (or drifts) in the data distribution with predictive accuracy approaching that of a traditionally trained model as the model trains more data.

The incremental models implemented, given incoming streams of observations, evaluate the model, detect drift, train the model and generate predictions, and in this project we will have their predictive performance measured with the metric MAE.

In the context of our problem, the model should be capable of learning the waiting times of Hospital Santa Maria, by continuously collecting data, updating the model by training it on the incoming observations in order to adjust the waiting times and generate predictions.

Method	MAE
BaggingRegressor	20.44
HoeffdingTreeRegressor	17.13
HoeffdingAdaptiveTreeRegressor	19.76
LinearRegression	16.84
SGTRegressor	15.93
EWARegressor	65.83

We tried different methods to build the model such as: LinearRegression, AdaptiveRandomForestRegressor, BaggingRegressor, EWARegressor, HoeffdingAdaptiveTreeRegressor, HoeffdingTreeRegressor, SGTRegressor, etc.

The model with the best predictive performance, in order words, with the lowest mean absolute error (MAE) was the following, using SGTRegressor:

```
model = compose.Select('Emergency_Stage'
    ,'lag_Waiting_Time_30_mn','
    People_Waiting')
model += (get_hour_and_weekday |
    feature_extraction.TargetAgg(by=['
        Service', 'hour', 'weekday'], how
        =stats.Mean()))

#model |= preprocessing.StandardScaler()
model |= tree.SGTRegressor(
     delta=0.01,
     lambda_value=0.01,
     grace_period=20,
 )

metric = metrics.MAE()
model
```

With a MAE of: 15.93 minutes.

The predictive performance of the different methods is ranked below:

We added the information we believed the waiting time depended on, in order for the model to take advantage of all possible information. We also applied a feature transformation to the Acquisition time, to get the hour of the day and the day of the week from monday to friday, adding more information to the model, thus decreasing the error:

```
def get_hour_and_weekday(x):
    x['weekday'] = float(x['
        Acquisition_Time'].weekday() < 6)
         # Mon:0,Tue:1,...,Sat:6,Sun:7
    x['hour'] = x['Acquisition_Time'].
        hour
    return x
```

Conclusion

To conclude our work, after doing some exploratory data analysis to analyse de hospital data using visual techniques, after applying drift detectors to the dataset to see how the statistical features of the target variable that the model is attempting to forecast vary over time in unexpected ways, and after applying incremental and online learning to build and train a model and generate predictions, we are able to answer the research question of predicting waiting times in different urgency services and at different emergency stages at a given point in time in the hospital Santa Maria, with a deducted prediction of 15.93 minutes.

References

[1] Journal - https://www.theportugalnews.com/news/2022-05-03/hospital-waiting-times-concerning-portuguese/66794

[2] Anton Pak, Brenda Gannon and Andrew Staib, Predicting waiting time to treatment for emergency department patients - https://doi.org/10.1016/j.ijmedinf.2020.104303

[3] Data set - https://drive.google.com/drive/folders/1tVyNhfzrRyXSxamILeQCgdW3JcOoGRZ0

[4] MathWorks, Incremental Learning Overview - https://www.mathworks.com/help/stats/incremental-learning-overview.html

[5] River Documentation - https://riverml.xyz/dev/

[6] River PageHinkley Documentation - https://riverml.xyz/dev/api/drift/PageHinkley/

[7] Class Lectures and resources, Streams Processing 2022