Predictive Analytics for Left Without Treatment in the Emergency Departments

Harish Kumar[1], Shahram Rahimi[1], Sean Bozorgzad[2], Alexander Sommers[1], Alexander Stephens[2]
[1]Computer Science and Engineering, Mississippi State University, Starkville, Mississippi, USA
[2]Potentia Analytics Inc., Carbondale, Illinois, USA

Abstract— *When patients arrive in a hospital's Emergency Department (ED) they are assigned an Emergency Severity Index (ESI) score that indicates the severity of their condition and the type of resources they may need. Patients with lower ESI scores are usually sicker, require more immediate service, and are more likely to require multiple resources. Patients who perceive their wait time as unacceptable may chose to leave the department at any time during their visit [1] [2]. Leaving the emergency department without receiving complete care is both a risk and quality of care issue. The work presented here seeks to produce a regression model which can predict the likelihood that a given patient will leave after having waited a specific amount of time in the emergency department. Such a model could be used to optimize the patient queue of an ED in an effort to minimize the likelihood of patients leaving without receiving care.*

Keywords: emergency department, hospitals, flow optimization, regression analysis, left without treatment, left without being seen.

1. Introduction

A typical emergency department serves patients who arrive without appointments at variable times with variable acuity of illness. Upon arrival a patient undergoes a triage process during which an ESI score is assigned to that patient. If the ED's rate of service is exceeded by the influx of patients, the patient is inserted into a queue, and is asked to wait given that their condition is not deemed urgent or emergent. Usually ESI scores 4, 5, and a select number of ESI 3 patients fall in to this category. The patient's position in the queue is dynamic and depends on the ESI score of all patients in the queue and how long each patient has waited for treatment. Those patients who receive treatment will be referred to as having "remained" and those that leave without being seen will be referred to as having "left" (or "LWOT", meaning "left without treatment").

The data presented here is a private dataset of approximately 65,000 patient encounters in the emergency department of an academic hospital in the US over a one year period. We have attempted to answer the following questions for each ESI level:

1. What is the probability that a given patient will leave having waited m minutes for treatment since arrival?

Table 1: Data-set broken down by patient type

ESI Level	Patients Remained	Patients left
1	975	0
2	19399	120
3	32173	1485
4	9698	517
5	614	54

2. What is the wait time with the peak probability of leaving without treatment (LWOT)?

To answer these, we must first answer two simpler questions for each ESI:

1. What is the distribution of wait times among the patients who elected to leave?

2. What is the distribution of wait times among the ones who remained?

We segregated the dataset, as will be described, and applied regression analysis to produce predictive models to predict a patient's likelihood of leaving. The findings of this work could, in future, be used to prescribe an optimal order of patients in the aforementioned queue.

The rest of this paper is as follows: Section II describes the segregation of the dataset, how the wait time was calculated for those who left and those who remained, any significant discoveries we made post-segregation, and the formula used to produce the probability of a patient leaving after having waited m minutes for treatment. Section III describes the probability curves produced from the segregated dataset, the regression analysis methods we applied to these curves, and gives a brief description of each method for readers unfamiliar with them. Section IV describes the means of testing, and reports the performance of each method applied. Finally, section V concludes the paper and suggests possible future directions.

2. Cleaning, Segregating and Processing of the Dataset

After removing all incomplete records, we were left with approximately 95% of our original dataset, the distribution of which is shown in table 1. In this work, we use the term "w-time" to refer to the time a patient waited before her/his visit was "resolved", either by being seen by a doctor, or by

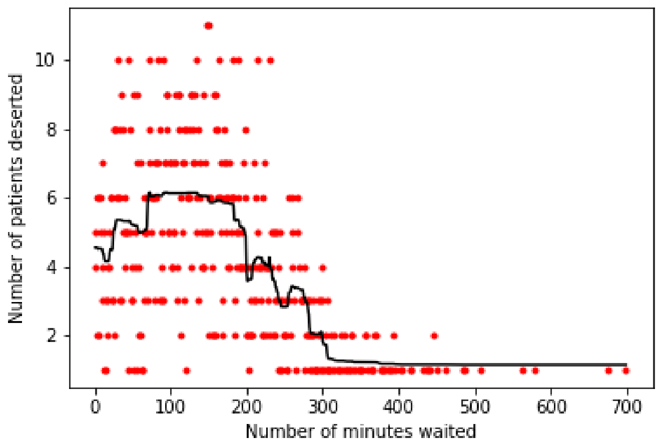

Fig. 1: Patients left at each minute at ESI level 3

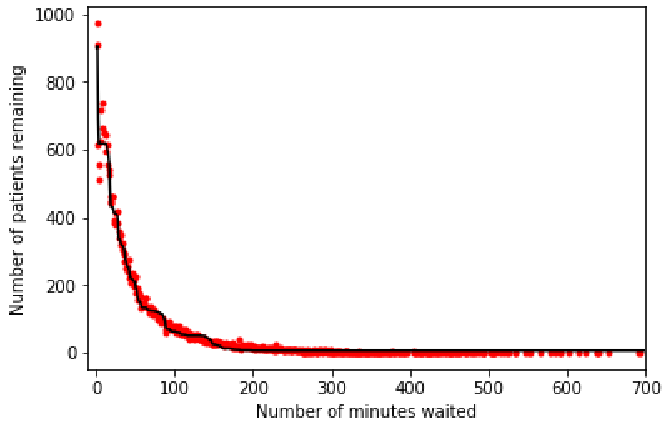

Fig. 2: Patients remaining at each minute at ESI level 3

Table 2: Approximate peak leaving time per ESI level

ESI Level	Approximate peak leaving time
2	67 minutes
3	133 minutes
4	108 minutes
5	85 minutes

Table 3: Longest w-time likely per ESI level

ESI Level	Longest w-time likely
1	20 minutes
2	40 minutes
3	60 minutes
4	75 minutes
5	65 minutes

leaving the ED. w-time is not an attribute in the dataset, but is easily derived from the provided attributes.

When a patient leaves without treatment, the w-time is simply the time of departure minus the time of arrival. When a patient remains, the first step could be assigning the patient to a bed or having a doctor examine them. The first step of this process, whatever it is, counts as the patient's time of "admittance" and the w-time for a remainer is the admittance time minus the arrival time.

Figure 1 shows a visualization of the ESI level 3 LWOT subset with a line of best fit superimposed on it (how this line of best fit is produced is discussed in the later sections), notice the peak at a w-time of approximately 133 minutes. The range around this peak contains the most frequently occurring w-times for LWOTs for ESI 3 patients. Each ESI level for the LWOT subset had such a peak, an approximate of which is shown in table 2 except for ESI level 1. ESI level 1 patients are critical patients and no record in our dataset shows any of them leaving without treatment.

Figure 2 illustrates the ESI level 3 remainer subset. Once again, there is a salient value, in this case the elbow point located at approximately 60 minutes. A vast majority of remainers at this level were seen with w-times at or below this point and thus this is likely to be the largest w-time any given ESI level 3 patient could be expected to have before being seen. The elbows of this kind in the other ESI levels' remainer subsets are seen in table 3.

Using equation 1, seen below, we have calculated the percent likelihood of a patient leaving for all ESI/w-time pairs that exist in our dataset.

$$P(D_{L,W}) = \frac{d_{L,W}}{(d_{L,W} + r_{L,W})} \tag{1}$$

$P(D_{L,W})$ is the probability that a patient of ESI level L, with a w-time of W, will leave. $d_{L,W}$ is a count of observed patients of that ESI level and w-time that did leave. $r_{L,W}$ is a count of observed patients, with those same characteristics, which remained.

So for example, if, for a given w-time and ESI level, there were 7 LWOT and 77 remainers the probability calculated by the above formula would be 0.08, quite low. This operation is committed for each w-time seen in each of the ESI level subsets, excluding ESI level 1.

3. Regression analysis

The above operations produce a derived subset for each ESI level. For each derived dataset the independent variable is the w-time and the dependent (target) variable is the probability of a patient leaving. It is to these derived subsets that our regression analysis was applied. In general, and by way of an introduction for the uninitiated, regression analysis is a kind of statistical modeling where one or more independent variables are used to predict one dependent, target, variable.

Being so well studied, we were able to select a number of off-the-shelf implementations of regression analysis methods to evaluate for the best performance on our data. The Orange

Table 4: Data-set broken down by patient type

Regression Algorithms	Accuracy of ESI 2	Accuracy of ESI 3	Accuracy of ESI 4	Accuracy of ESI 5	Overall Accuracy
Linear Regression	0.681	0.605	0.560	0.142	0.497
Decision Tree Regression(max-depth:1)	0.773	0.904	0.787	0.04	0.626
Decision Tree Regression(max-depth:3)	0.949	0.971	0.925	0.20	0.761
Decision Tree Regression(max-depth:5)	0.950	0.986	0.949	0.19	0.768
Random Forest Regression(max-depth:1)	0.859	0.905	0.800	0.20	0.691
Random Forest Regression(max-depth:3)	**0.958**	0.977	0.947	0.22	**0.775**
Random Forest Regression(max-depth:5)	0.951	**0.988**	**0.951**	0.16	0.762
K Nearest Neighbours(n_neighbours:50)	0.956	0.986	0.937	0.176	0.763
SVR	0.788	0.730	0.712	0.09	0.58
Isotonic Regression	**0.958**	0.984	**0.951**	0.255	**0.787**
AdaBoost Regression	0.957	0.972	0.940	**0.285**	**0.7885**

Table 5: Data-set broken down by patient type

Regression Algorithms	Accuracy of ESI 2	Accuracy of ESI 3	Accuracy of ESI 4	Accuracy of ESI 5	Overall Accuracy
Linear Regression	0.749	0.561	0.534	0.077	0.480
Decision Tree Regression(max-depth:3)	0.930	0.980	0.935	-0.066	0.696
Random Forest Regression(max-depth:3)	0.936	**0.984**	**0.943**	0.104	**0.741**
K Nearest Neighbours(n_neighbours:50)	**0.951**	0.979	0.915	0.188	**0.758**
Neural Network	0.925	0.964	0.924	**0.281**	**0.773**
AdaBoost Regression	0.899	0.975	0.914	-0.357	0.607

data mining toolkit [20], and the python library “Scikit-learn" [21] provided us with the implementations of the selected regression models, each of which is presented below with a brief description.

3.1 Linear Regression

Linear regression prediction models assume a linear relationship between the predicted (dependent) variable(s) and the predicted from (independent) variable(s). Simple linear regression has only one independent variable while “multiple" linear regression has several. The general form of linear regression can be expressed with the following function:

$$y = h(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n \qquad (2)$$

Where y, the predicted value, is predicted by calculating the weighted sum of x_1, x_2, ..., x_n, each of which is multiplied by some weight theta [3] [4].

3.2 Decision Trees

This method can be used to produce regression predictors though it is more commonly known for producing classifiers. On the basis of its attributes, a record will traverse down the tree, being sorted at each junction of branches on the basis of a splitting rule. A record is classified or a real-valued prediction is made for it on the basis of which terminal, “leaf", node it is finally sorted into. We varied the “maximum depth" hyperparameter in this work to examine the effects this had on performance. This max depth value determines the maximum number of sorting rules that can exist in the longest such path down a tree. If this number is too high for a given dataset then over-fitting is more likely and the tree may generalize to new data poorly. We used max depths of 1, 3, and 5 for our experiments [5] [6].

3.3 Random forests

Random forests are several different decision trees generated using the same data-set. This ensemble of trees produces predictions by having the predictions of the individual trees aggregated using a variety of methodologies [7] [8] [9].

3.4 Support Vector Machine

The version of the support-vector machine (SVM) used for regression is called a support-vector regression (SVR) model. There is a hyperparameter for this model called its “margin". The margin is a distance about the line which will come to be the regression line formed via training. Points within this distance are those used to calculate the error of the model, and it is on the basis of this error that the line of best fit is adjusted. Because of the margin, only a subset of the dataset is used to form the final regression model in SVR [10] [11].

3.5 K-Nearest Neighbors

In the K-Nearest Neighbor (KNN) method a data-point for which some value is to be predicted is projected into the already possessed dataset, the points of which have all their values known. The k nearest points to this projected point, its k nearest neighbors, are used to calculate the new point’s predicted variable value. This is most often done by averaging the salient variable value of these k neighboring points. The hyperparameter k is obviously of importance and selection of a good k is a problem on its own. We used a k value of 50 neighbours for this problem [12] [13].

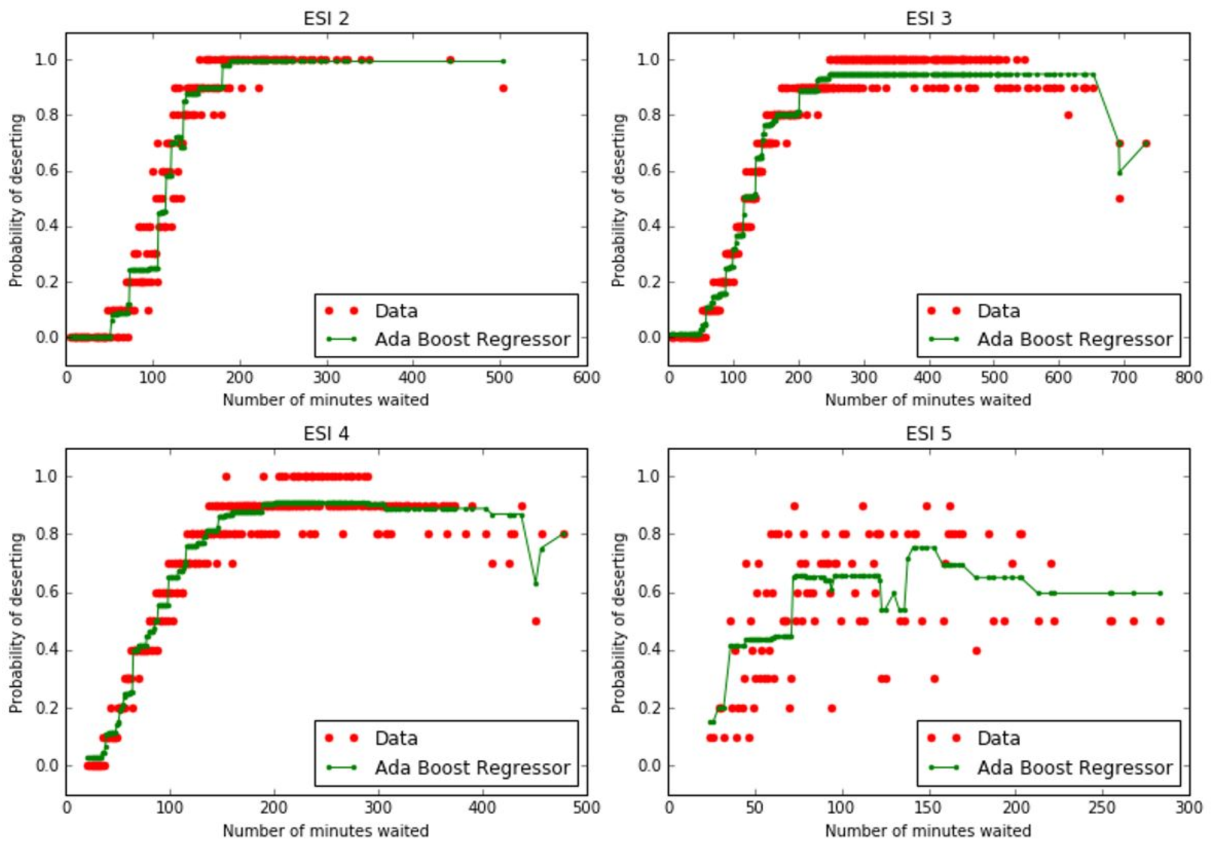

Fig. 3: Adaboost Regression

3.6 Isotonic regression

In this approach, using at least a 2D attribute set, a line is fitted to the data with the following constraints: (1) The line must be as close to all points as possible, which amounts to minimizing the distance between the line and all data-points. (2) The line's trend must be monotonic, meaning that it must exclusively have a positive or negative trend, though segments are permitted to be flat, with a slope of zero [14] [15]. Following the above two constraints over multiple iterations, the system will converge to a solution.

3.7 AdaBoost

Adaptive boosting, or AdaBoost, is a kind of meta learning model. An ensemble of learning methods (such as those discussed above) is produced and the outputs of each of them are then the inputs to the meta model, which similar to standard regression, learns a series of weights. These weights are adjusted in a way to minimize error across the training data-set [16] [17] [18] [19].

4. Results and Discussion

Each regression model was trained with 80% and tested with 20% of the derived data. The accuracy of each regression method are then averaged across the ESI level datasets, to which it was applied, to get the final, overall, measure of their accuracy. Table 4 presents the accuracy of each method implemented with Scikit-learn, both on the individual derived ESI datasets and the aggregated values across all four ESI levels.

It can be seen that Scikit-learn's AdaBoost algorithm performed best overall, followed by Isotonic regression implementation. It is not completely surprising that AdaBoost performed better overall than any one random forest or decision tree since it uses ensembles of machine learning algorithms like them to perform a more sophisticated analysis.

Orange's implementations give us a different ranking for similar parameter settings. Here the neural network placed first, followed by KNN with k = 50 as the second best. Orange's results are shown in table 5.

The stochastic nature of the generation of the training and the test sets is undoubtedly responsible for some variation of performance among analogues regression models when comparing the Orange and SciKit-learn results, but we suspect there is more to the variation than this alone.

A KNN implementation in each suite had k = 50 and the AdaBoost instances both had 100 estimators so at least these hyperparameters cannot be the source of the variation.

Investigating the reasons for these variations would require more study and, possibly, should be done by independently developing the above algorithms outside the aforementioned packages. As important as they may be, these tasks do not fit in the scope of this paper.

Without a deeper knowledge of the architecture of the ambiguously titled “neural network", more thorough knowledge of the sub-learners used by both suites' AdaBoost implementations, and an execution of a k-fold cross validation for all experiments, our only response to the ambiguity as to which regression models is superior is to appeal to raw numbers: Scikit-learn's AdaBoost has the highest overall performance of any fielded implementation. Figure 3 illustrates the regression curves produced by Scikit-learn's implementation of AdaBoost for each ESI level.

5. Summary and Future Work

In this work we evaluated several different regression methods and successfully, if somewhat provisionally, identified the best models for prediction of the likelihood of a patient leaving the emergency department without being seen, given their ESI level and w-time.

As to future work, now that the highest performing regression method is revealed (of those we selected to test), tentative as this declaration of superiority is, we can easily imagine applying the predictions of an ensemble of trained classifiers, one for each ESI level, to the optimizing of an ED's queue. The queue in an ED could be dynamically reordered whenever a new patient arrives, and as the waiting time of each patient increases, to minimize the net probability of LWOT ratio.

Other countermeasures to LWOT might also be informed by such predictors. For example: When a patient's leaving probability becomes too high a staff member of the hospital could reassure them of the shortness of their coming wait, remind them of the severity of their condition and the danger posed to them by leaving before being seen, or both.

References

[1] Weiss SJ, Ernst AA, Derlet R, et al. Relationship between the National ED Overcrowding Scale and the number of patients who leave without being seen in an academic ED. Am J Emerg Med.2005; 23:288-294.

[2] Asaro PV, Lewis LM, Boxerman SB. Emergency department overcrowding: analysis of the factors of renege rate. Acad Emerg, Med. 2007; 14: 157-162.

[3] GÃl'ron, A. (n.d.). Linear Regression - Hands-on machine learning with Scikit-Learn and TensorFlow.

[4] Linear regression. (2019, June 15). Retrieved from https://en.wikipedia.org/wiki/Linear_regression

[5] 1.10. Decision Trees. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/tree.html#tree

[6] Decision Tree Regression. (n.d.). Retrieved from https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html

[7] Random forest. (2019, May 30). Retrieved from https://en.wikipedia.org/wiki/Random_forest

[8] 3.2.4.3.2. sklearn.ensemble.RandomForestRegressor. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[9] Hewa, K., Hewa, K. (2018, November 27). A Beginners Guide to Random Forest Regression. Retrieved from https://medium.com/datadriveninvestor/random-forest-regression-9871bc9a25eb

[10] Saedsayad.com. (2019). Support Vector Regression (n.d.). Retrieved from https://www.saedsayad.com/support_vector_machine_reg.htm

[11] GÃl'ron, A. (n.d.). Support Vector machine - Hands-on machine learning with Scikit-Learn and TensorFlow.

[12] K-nearest neighbors algorithm. (2019, May 27). Retrieved from https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[13] Sklearn.neighbors.KNeighborsRegressor. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html

[14] Isotonic regression. (2019, May 11). Retrieved from https://en.wikipedia.org/wiki/Isotonic_regression

[15] Isotonic Regression. (n.d.). Retrieved from https://scikit-learn.org/stable/auto_examples/plot_isotonic_regression.html

[16] AdaBoost. (2019, May 29). Retrieved from https://en.wikipedia.org/wiki/AdaBoost

[17] Sklearn.ensemble.AdaBoostRegressor. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostRegressor.html

[18] Brownlee, J. Boosting and AdaBoost for Machine Learning. (2016, September 22). Retrieved from https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/

[19] Towards Data Science. (2019). Understanding AdaBoost. (n.d.). Retrieved from https://towardsdatascience.com/understanding-adaboost-2f94f22d5bfem/

[20] Demar, J., Curk, T., Erjavec, A., Hoevar, T., Milutinovi, M., Moina, M., Zupan, B. (1970, January 01). Orange: Data Mining Toolbox in Python. Journal of Machine Learning Research 14(Aug):2349âĹŠ2353. Retrieved from http://jmlr.org/papers/v14/demsar13a.html.

[21] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.