MedClick: Last Minute Medical Appointments No Show Management

Daniel Sousa
daniel.monteiro.sousa@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal

November 27, 2017

Abstract

A no-show occurs when a client has an appointment of some sort with another entity and, voluntarily or not, the client does not show up to that appointment. This termwas originally popularized by airline companies, but since then it has also been used by hotel companies and the health care sector, where these occurrences are extremely harmful. A patient missing an appointment will mean that the clinic's and health professional's time slot will be wasted. The goal of this article is to find a solution that minimizes no-shows, detecting when a patient is not going to come to the appointment and finding an appropriate replacement. There are already some implemented solutions to help with the no-show problem, mainly focused on machine learning techniques, that are used to find a time slot where there is a risk of occurring a no-show so that the clinic can overbook accordingly. The outcome of this article and related research is an algorithm that, instead, can detect an appropriate replacement for that appointment. The solution that was developed during the execution of this project is a hybrid solution which combines two different behavior prediction techniques: population based behavior and individual based behavior. The algorithm starts by computing a no-show probability based on the population's behavior using a logistic regression model. After that, using Bayesian Inference, that probability is personalized to each patient. After computing the no-show probabilities for every candidate patient, the algorithm checks if any of them are interested on taking the appointment. The algorithm's results were fairly positive, showing a generally expected behavior for all the datasets that were provided to it.
Keywords: No-Show, MedClick, Appointment, Bayesian Inference, Logistic Regression, Hybrid Approach

1. Introduction

A no-show is a term widely used to describe a client who has an appointment of some sorts with another entity and, voluntarily or not, the client does not show up to that appointment. This term was originally used by airline companies, when they started to notice that this behavior was more common than they originally thought, thereby creating a specific name for the phenomenon. Since then, the term has been used by other businesses, such as hotel companies and the health care sector. Although a no-show is always a negative influence on its respective business, the health care sector is where it is the most harmful, for various reasons. A patient that misses an appointment, or just doesn't arrive on time, will waste that clinic's and health professional's time slot, which could otherwise be used on another patient.

This article will be focused on developing a no-show algorithm. The objective of this algorithm is, based on the history of appointments of several health care centers and using several parameters from patients' data, to build a model that is able to make reliable predictions of a patient's future behavior. This should take into account the behavior of the whole population on the platform, but also take into account the specific behavior of the patient in question, as this last behavior can change drastically the probability of no-show. After this model is built, the algorithm and the platform should be responsible for contacting and notifying patients, as to know whether or not they are coming to the appointment. If the system detects that they aren't coming (patient does not respond to notification, for example), the algorithm should find a replacement, using the model described above. When the algorithm ends, whether it found a replacement or not, the appointment information should be updated accordingly, and the health care provider should be notified of the change.

The rest of this document is organized as follows. Section 2 will talk about population based approaches, section 3 will cover individual based approaches, section 4 will cover how both ap-

proaches can be combined, section 5 will cover the developed solution, section 6 presents the results of the evaluation done to the algorithm and finally section 7 summarizes the work done and describes possible improvements.

2. Population based approaches

A population based method is a method that considers the entire behavior of the population and uses that data to make its predictions. Normally these methods are based on statistical models and machine learning algorithms. Common algorithms used for these methods are Decision trees [14, 4, 12], k-Nearest Neighbors [8] and rule-based models [1, 18, 10, 6]. This section will cover a statistical model called Logistic Regression.

The logistic regression model is a regression model that is mainly used, based on a set of predetermined features, to predict problems where the outcome is not continuous, but discrete [11], normally binary. An example application of this model would be a problem where we want to determine if a tumor is malignant based on its size. The result in this case would only be Yes or No. The features received by the algorithm can be continuous or discrete, but are always represented numerically (for example, for the no-show algorithm, one of the features could be the patient's sex, where a male patient would be represented by 0 and a female patient represented by 1).

This statistical model is very similar to a Linear Regression model, where the outcome is a continuous variable [16]. Contrary to a Linear Regression model, which uses the linear equation to make its estimations ($y = mx + b$), a Logistic Regression model uses the Logit function:

$$F(x_1, x_2, ..., x_k) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + ... + \beta_k x_k)}} \quad (1)$$

where k is the number of features used in this model, x_1 to x_k are the features and β_0 to β_k are the coefficients, each corresponding to a feature except β_0, which is the independent coefficient. These coefficients are what shape the logit function to fit the problem.

This function has the special characteristic of being upper-bound by 1 and lower-bound by 0 on the y axis. As these are normally problems with binary results, this is very advantageous, as we can consider the y axis as a probability of the problem happening, as seen on Figure 1. The example represented has only one feature (the X axis) but the same method can be applied to multiple features, on a k-dimensional plot.

This algorithm has two main phases: model building and prediction. The model building phase estimates the best coefficients that would fit the given problem, while the prediction phase uses those coefficients on the logit function to make the needed predictions.

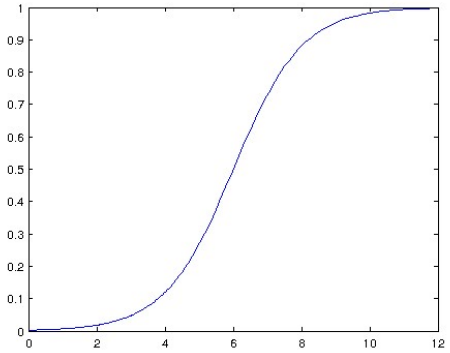

Figure 1: Example of Logit function.

The model building phase is where the algorithm will estimate the coefficients, to try to give the most accurate model possible for the prediction. As such, a cost function needs to be built. To do that, we need to make several estimations of the coefficients. Then, using the residual sum of squares, represented in equation 2, we can build the cost function where, the bigger the error is, the worse the estimate is. The equation is represented below, where m is the number of test cases (which corresponds to the size of our dataset), $E(X_i)$ corresponds to the estimation (solving equation 1) and y_i corresponds to the real value.

$$J(\beta_0, \beta_1, ..., \beta_k) = \sum_{i=1}^{m} (E(X_i) - y_i)^2 \quad (2)$$

After that, we use the gradient descent algorithm which can be used to find the global minimum of that function, giving us the estimated best coefficients for that problem. The gradient descent algorithm will rarely find the perfect coefficients, as there are problems that don't have a function that can perfectly describe it (because data in real life does not always have a clear pattern and there are always outliers present in it), but the coefficients are accurate enough for an estimate.

The prediction phase occurs after the model is built, and it uses the estimated coefficients and the logit function to make a prediction, based on a set of provided features. This is simply done by replacing the appropriate values on the logit function and computing the probability.

Logistic regression is widely used to solve no-show problems because it simply requires a set of features from the patients and it can reliably predict their future behavior [2, 5, 7]. The main advantage of this algorithm is its speed as, despite the model building being computationally expensive (lots of estimations needed to have an accurate model),

after being built, computing the probability of a patient missing his appointment is extremely fast, as it is only needed to make a simple computation. Also, in case it is needed to add or remove new features to the model, it is very simple to do so, as it is only needed to rebuild the model with the updated set of features and it will be ready to use.

The main disadvantage of this, and mainly all other machine learning and statistical algorithms, is that it needs a large dataset to be precise and the features that are used for the algorithm need to be previously studied, as features that do not affect the result can greatly affect the precision of the logistic regression model.

3. Individual Based Approaches

Individual based approaches use past behaviors of an individual to make an estimation, contrary to using the behavior of the whole population. While these types of algorithms are a fast and easy way to model the behavioral pattern to each individual and work really well with a relatively small dataset, they are also extremely dependent from the individual's past history to make accurate predictions, and as such should be used when we are certain that we have enough data to make an accurate estimation. There are various examples of these approaches, such as Moving Average algorithms and Hidden Markov Models [9, 15, 17, 13, 3], but what this article will cover is Bayesian inference.

Bayesian inference is a method of statistical inference that is used to update a certain probability, using information that was meanwhile obtained. This type of statistical model is based on the famous Bayes' theorem (Equation 3), which on itself is used to determine a probability of an event based on prior knowledge of conditions that might be related to that event:

$$P(A|B) = \frac{P(B|A)}{P(B)} \tag{3}$$

where A and B are events. As such, this method can be used to update a prior probability to a posterior, based on new information. This method is used in [2] to update the probabilities obtained previously with Logistic Regression. The authors were considering three different events (no-show, cancellation and show-up) and as such, since the data is more complex than a simple set of probabilities, the original Bayes theorem needs to be adapted to the problem, and in this case, it was used the following equation, which is an adapted posterior mean:

$$E(a_k|y_1, ..., y_k) = \frac{y_k + a_k}{\sum_{k=1}^{K} y_k + \sum_{k=1}^{K} a_k} \tag{4}$$

where K corresponds to the number of events (in the article's case, 3 different events), a_k corresponds to the corresponding probability of the event k and y_k corresponds to the number of occurrences of the event k in the database.

This means that to successfully update the no-show probability with this method, all that is required is the count of each event and the respective previous probability, which means that a simple and fast query would get the needed information. While this is a simple method to use, it is efficient as the probability updates accordingly to the situation (Table 1).

4. Hybrid approach

The solution that was ultimately implemented was a hybrid approach, based on the research done in [2]. In that research, it was combined both a population based method and an individual based method, logistic regression and bayesian inference respectively. As such, their final solution starts by using a logistic regression model to capture the general behavior of the dataset, using previously studied parameters, such as sex, age, marriage status, among others. This model is used to compute an initial estimation of the no-show probability, using the general behavior of the population. After that, it is used the posterior mean equation (equation 4) of the bayesian inference method to adapt a patient's probability. As such, this hybrid approach becomes a more versatile and accurate solution than only using one of the methods.

5. Solution

This section will focus in the description and implementation of the no-show algorithm, using the previously described methods. The developed algorithm is a hybrid solution based mainly on the solution developed in [2], using both population and individual based approaches, where it starts by building a model which predicts general behavior and later personalizes the predictions based on the individual's personal history, if there is one. As such, the main solution is divided in two different sections, both explaining the chosen approach and the reason for its choice.

5.1. Model Building

For the initial phase of model building and modeling a population based on their behavior for prediction, logistic regression was chosen, as there are various no-shows solutions already implemented using this statistical model, with very good results in general. As described above, the proposed solution firstly uses a logistic regression approach to its model building. As previously described in section 2 and represented in equation 1, to build a logistic regression model it is needed a dataset to train it

Table 1: Attendance record and Bayesian update probabilities

Appointment No.	Appointment date	Attendance record	no-show	Cancellation	Show-up
			0.14	0.09	0.77
1	10/5/2009	1	0.57	0.05	0.38
2	10/29/2009	1	0.71	0.03	0.26
3	11/5/2009	2	0.53	0.27	0.19
4	12/4/2009	1	0.63	0.22	0.15
5	12/7/2009	1	0.69	0.18	0.13

and a set of features in order to describe it. In this context's problem, the dataset that is needed is the appointments' history.

As for the set of features, due to some data limitations, the features that were chosen were based on the ones used in [2] (patient's age, patient's sex, appointment date and patient's distance to the health care center). These features were chosen because of two main reasons. The first being this was easy to obtain information, only requiring some basic information from the patient and the health care center. The second reason is that these features were studied to have an impact on the patient's behavior to show up or not to an appointment, and as such were good candidates to use for the model. However, the solution developed in [2] was developed in an health care center in another country, and not in Portugal, and as such there was a chance, before testing, that these features could not be as significant in Portugal. Unfortunately, the company could not provide in time real world data in order to be studied and to determine which features would be the best fit for that dataset. As such, it was decided to follow those set of already studied features to be used in our no-show algorithm, with the possibility to add more or replace them in the future, in case its necessary.

When the platform is launched, the model is automatically built using all the appointment data available. As previously explained, the output of the logistic regression algorithm is a set of coefficients, which correspond to each of the patient's features, which will be saved when the model is built. This set of coefficients will then be applied to equation 1 with the features of the appointment that we are trying to predict in order to get the estimated no-show probability. As such, after the model is built, we only need to input a list of features and the algorithm will quickly output the respective no-show probability. Since new information will arrive in the system constantly, the model will need to be updated frequently, as to update the coefficients. This is easily done by rebuilding the model. Since it's not efficient to rebuild the model every time new information arrives in the system, the model is rebuilt only after a certain number of records have been inserted in the database (that number can be customized). This way, the model will continuously have updated information and as such will be able to give more accurate predictions. The simplified view of this whole process is represented in Figure 2, which is repeated for every 10 000 records (by default).

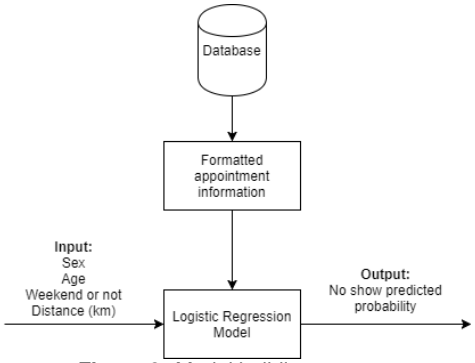

Figure 2: Model building process.

5.2. Profile personalization

After the initial probability, based on the general behavior of the population, is computed, the algorithm needs to adapt that probability to the specific patient, based on his appointment record. To do this, the equation based on bayesian inference from [2] was used (equation 4). The reason for this choice is that, compared to the other methods, it is a very simple method to use, with no data structures needed to build and fast computation. After computing the initial no-show probability, the system makes a simple query on the database that retrieves all the appointments made by the candidate patient. Then, the system counts the total number of appointments made and the number of appointments where the patient made a no-show. This is all required to use on equation 4. After that, these values are used in that equation, giving us the updated no-show estimated value for that specific patient.

5.3. Algorithm execution

After describing each main component of the no-show algorithm, we can now start to describe the no show algorithm as a whole, explaining how it works from the moment it receives a no-show notification until it is able to find, or not, an appropriate replacement for the appointment. The whole algorithm is represented in a flow diagram, which is shown in Figure 3.

The algorithm starts by receiving a no-show no-

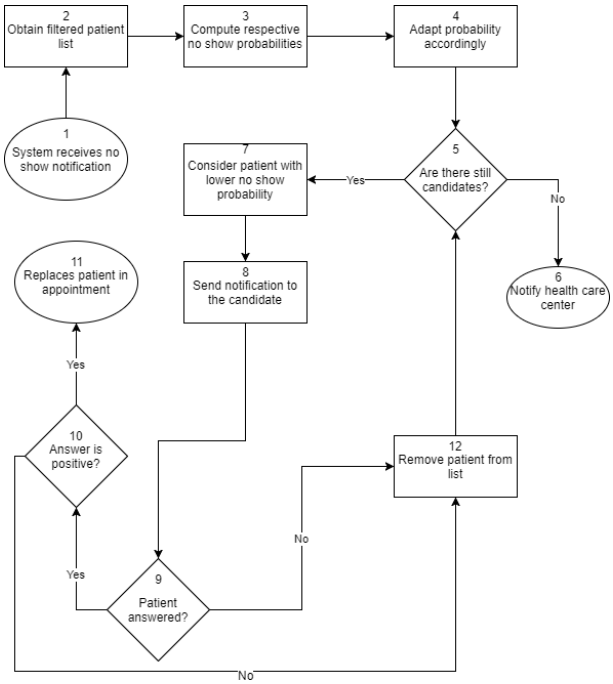

Figure 3: no-show algorithm flow diagram.

tification, which can happen in three different situations. The patient can cancel its appointment, the patient can fail to respond to an appointment confirmation or the system will detect that the patient will not arrive on time (by the distance to the health care center). If the former two situations happen, the patient will be penalized in its no-show rating, but if the patient just cancels the appointment its no-show rating will not be affected. After that, the algorithm obtains a filtered patient list, which is comprised of two different lists.

The first one is the appointment's waiting list, which will have all the patients who have an appointment scheduled in the future, starting from the same time that is left to the appointment (for example, if the appointment we are trying to fill is one week from now, the list will only consider patients with an appointment further than one week from now). This is done to avoid having patients missing the opportunity to receive the notification, in case the previously considered patients take too long to respond. The second list will be comprised of all patients within a certain distance from the health care center. This distance can be customized by each health care center, as depending on their locations, different values would be appropriate.

These two lists are considered in order, as all patients from the first list will be considered before the patients from the second list. This is done to give priority to patients who already have scheduled an appointment of this type, being more likely to accept a change in the date of the appointment.

After computing the lists, the algorithm will compute the no-show probability for each patient on both lists, using the logistic regression model previously built to compute the general probability and using the posterior mean equation to adapt the probability for each patient. After this step, each patient will have an associated no-show probability.

The algorithm then enters its main loop, where it starts by checking if there are any candidate patients left. It also checks for the time remaining to find a replacement. If the time to the appointment is less than the time given for the patient to respond to the notification plus an additional time of two hours (by default, can be customizable by each health care center). This additional time is added as a worst case scenario, as to give an extra time between the acceptance of the last candidate and the time for the candidate to prepare and go to the health care center.

If there are no patients in neither of the lists, then the health care center is notified that the algorithm was unable to find an appropriate replacement. Otherwise, the loop continues the iteration by considering the patient with the lowest no-show probability. As previously stated, the algorithm will firstly choose patients from the waiting list and only after that will it go to the other list. After that, the system will send a notification to the candidate, which can be done by e-mail, mobile app push notification or sms. This is defined by the patient, but currently, due to API limitations and the mobile app not being implemented yet, the only available notification is by e-mail. The patient will have a deadline of 12 hours to respond. If the patient responds positively, the algorithm ends by replacing the time slot of the appointment with the new patient. However, if the patient responds negatively, or, after 12 hours, does not respond, the algorithm will move on to the next patient, while removing the last patient from the filtered list.

6. Evaluation

The no-show algorithm was evaluated with three different main tests. Firstly, a time evaluation, to determine if the model building was efficient with a large amount of data. Secondly, with a list of scenarios, to test the behavior of the algorithm with different population behaviors. Finally, evaluation with real world data, provided by "MD Clinica".

6.1. Time Evaluation

In addition to evaluating if the algorithm produces the expected results, it is also important to consider its speed. Since the MedClick platform will be receiving a large amount of data everyday from multiple different health care centers, the no-show algorithm will be using a considerable amount of

data. As such, it is needed to confirm that the algorithm can scale well with the amount of data that is given to it. The main objective of this test is to give the algorithm continuously growing amounts of data and see how long it takes to build the logistic regression model. For the dataset, it was given to the algorithm random data with no patterns and patterned data. For each dataset given, the algorithm was fed a number of records ranging from 1000 records to 10 million records, and measured the build times of every 1000 records in that interval. That data was fed to a linear regression algorithm, which could give us the estimation of the build times for a higher number of records. This was also tested in a machine with a 2 core processor, and as such it was tested with 1 and 2 cores. The results are represented in Figure 4.

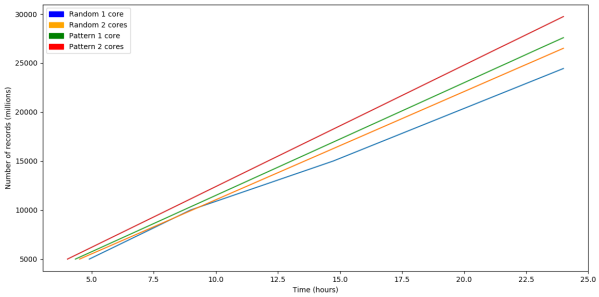

Figure 4: Times evaluation results graph.

We concluded that the algorithm will take a full day to build its model with an estimated number of around 24440 million records, which is a considerable amount of records. In the best case scenario, with data containing clear patterns, the algorithm will be able to process around 29738 million records. Since this is a considerable amount of data and since the algorithm will take some time to reach this amount of data volume, we can assume that the algorithm will build the logistic regression model efficiently.

6.2. Scenarios

For these tests, a mock database was created, which consisted of a small list of patients, one health care center and, depending on the scenario that was being tested, a list of appointments that represented the behavior that this specific scenario was trying to represent. For each scenario, the logistic regression model was built with the provided data. Also, for each scenario, the data was modified on its size and some of its characteristics, in order to show how the algorithm behaves with small or big changes in the data. For scenarios 1 to 4, that are testing the population based behavior of the algorithm, it was used data with 10, 100 and 1000 records and for each the scenario applies to 100%, 80%, 40% and 0% of it. Scenario 5 will have a base of 1000 records, with additional 10, 50 and 100 of only one patient, in order to test the algorithm's profile personalization.

Scenario 1 considered a binary parameter, the sex of the patient. For the population, it was considered the percentage of male patients that missed their appointments. The results showed that the algorithm behaves as expected, giving the expected results with as little as 10 records, showing that the algorithm picks up this patter with little data. However, for the 80% tests, the results are not as consistent, as they should be higher than the 40% tests. This may be due to this test only considering only one of the features, while ignoring other features like the distance, that varies greatly in some patients. The results of this scenario are represented in Figure 5.

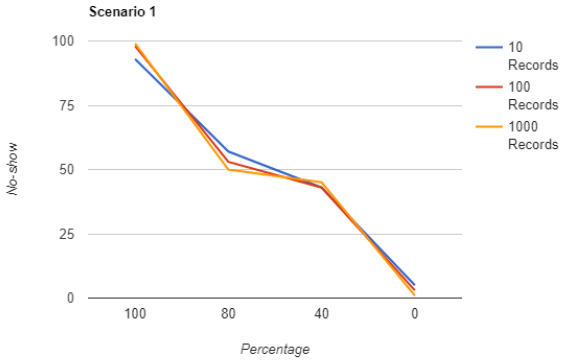

Figure 5: Scenario 1 test results.

Scenario 2 considered a continuous (non binary) parameter, the age of the patient. For the population, it was considered the percentage of patients over 25 years old missed their appointments. The objective of this scenario is to detect how well the algorithm can detect a pattern from an interval of values. In general, the results go according to what was expected. For all percentages, the algorithm starts by giving a good estimation with 10 records, improving with 1000 records. Also, as the percentage goes down, the probabilities also go down reasonably, more so with a higher number of records. One thing to note is that, due to this scenario evaluating an interval of values, in a lower number of records patients in the same interval but with a big difference of age had sparse no-show probabilities. For example, with 10 records, patients 7 and 8, which are right on the 25 year mark, and patient 6, which has a much higher age than the others, had very different, sometimes seemingly random probabilities, with a low number of records, which ultimately affected some of the results. These big differences disappeared, however, with a higher number of records, where there was

a clearer distinction between the patients with age near 25 and lower, which showed a very different no-show probability for patients with 25 and up, which was expected. The results of this scenario are represented in Figure 6.

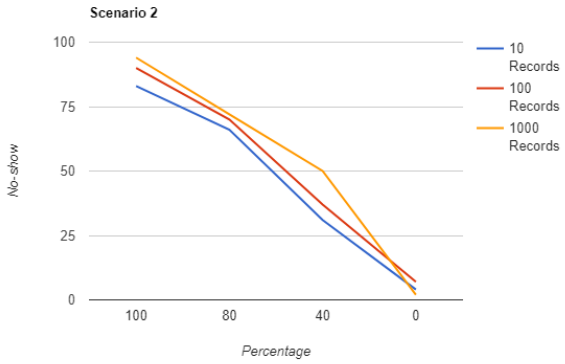

Figure 6: Scenario 2 test results.

Scenario 3 considered two continuous parameters, the age of the patient and the distance to the health care center. For the population, it was considered the percentage of patients over or equal to 25 years old and over 50 kilometers who missed their appointments. The objective of this scenario is to detect if the algorithm gives consistent results when there is a pair of parameters that affect the patient's behavior. In general, results go according to what was expected, although not as much as previous scenarios. For the 40% row, the predictions seem to have a more inconsistent behavior, as the probability should be tending for around 40%, and not straying away from it. However, the predictions are still fairly accurate. Also, for the 0% row, the no-show probabilities are still relatively high, even at 1000 records, when they should be tending to 0%. This shows that these types of more complex patterns need a considerable size of data in order for the algorithm to detect them. The results of this scenario are represented in Figure 7.

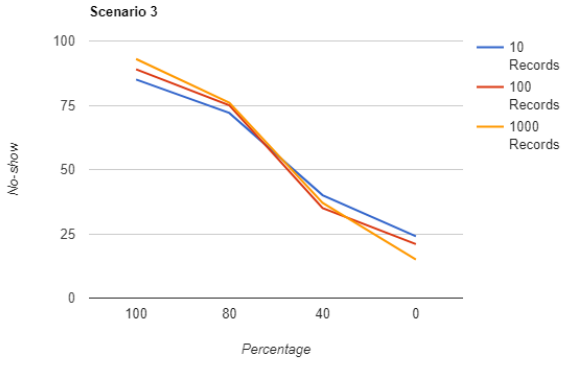

Figure 7: Scenario 3 test results.

Scenario 4 considered two binary parameters, the sex of the patient and the day of the appointment (weekend or not). The objective of this scenario is to compare the algorithm's behavior with scenario 3, in order to observe if there is any difference on the predictions based on the type of parameters. In general, results go according to what was expected, even if there are some small inconsistencies. In general, the algorithm starts by giving a fairly good no-show probability estimation with only 10 records. These initial predictions are also less accurate than the scenarios with only one parameter, which is to be expected. As the number of records go up, the predictions start to get better and more consistent than with the previous scenario. This shows that, while the algorithm handles well both types of parameters, making accurate predictions with large volumes of data, the algorithm recognizes the patterns in data quicker with binary parameters. The results of this scenario are represented in Figure 8.

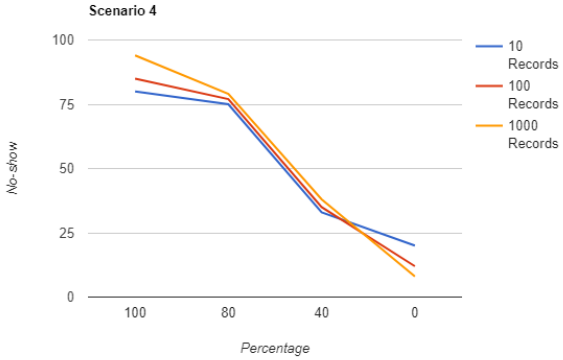

Figure 8: Scenario 4 test results.

The goal of this scenario is to check how the algorithm handles the profile personalization of a patient, considering the behavior of only one patient. It was considered a record size of 1000 records, all with a defined behavior of all male patients missing their appointments. Then, it was inserted new records of a male patient that has no records in the database, with the opposite behavior (attending the appointments). As such, it was considered a varying new records size of 10 to 100, with varying percentages each which correspond to the percentage of records where that new patient showed up to the appointment. The posterior mean equation was extremely efficient in responding to the patient's behavior. Even with 10 records, the algorithm gives a very good estimation of the no-show probability, while still being affected by the general population's behavior. This effect, however, becomes less and less prevalent with more and more records, which is the optimal behavior.

In the 0% percentage, the probability is the same in the whole row because the behavior of the patient matches the behavior of the general population, and as such the probability does not change from its initial prediction.

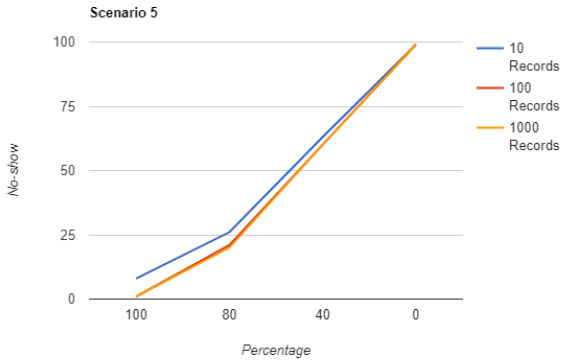

Figure 9: Scenario 5 test results.

What we have been able to conclude with this evaluation is that the devised algorithm is capable of picking up various patterns in data, either being simple patterns of only one feature or a combination of features. The logistic regression model, while sometimes having some difficulty on picking up patterns with a low number of records, the patterns are clearly defined with 1000 records. Since the algorithm will be dealing with a much larger number of records, this will not be a problem. For the posterior mean equation, it handles the personalization of the no-show probability very well, as demonstrated in scenario 5, as it responds to shifts in the patient's behavior very fast, while also taking into account the general behavior.

6.3. MDClinica data

The final set of tests that were done to the no-show algorithm were about the data received by MD Clinica. It is important to test the algorithm with this kind of data as the patterns in it are not as clear as the data that was used in previous tests, and as such we can see how the algorithm will behave in these situations. To evaluate the algorithm with this data, three different main tests were made. Model accuracy, feature behavior and patient history tests.

For the model accuracy, we divided the given data in two parts, the training data, that is the data that is given to the model, and test data, which the algorithm will try to predict. For each row in the test data, the algorithm gave a prediction on which outcome would be more likely (no-show or show up), and with that, the model's accuracy (equation 5) is computed:

Table 2: Accuracy test results

Test data	Accuracy
10%	0.73
20%	0.72
30%	0.7
40%	0.7
50%	0.68
60%	0.67
70%	0.67
80%	0.68
90%	0.67
100%	0.78

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{5}$$

For this test, it was considered a threshold of 50%, which means that the result that got a probability of over that value will be the result that is considered. These tests were also only done using the logistic regression model, and without the Bayesian inference portion of the algorithm, as the objective of this test in particular is to test the model's precision with this kind of data. The amount of the test and training data that was given to the algorithm was also varying, ranging from 10% of the data being training data to 100%, where the model was given all the data and that same data was given as training data. The results of this test can be seen in table 2.

These results show that the accuracy of the logistic regression model is generally good, considering that this test do not contain the profile personalization section and that some data may contain some erratic behavior from the patients, which is to be expected. With 10% and 20% of test data, which is often the used amount, the model reaches around 73% of accuracy, which lowers slightly as the training data diminishes in size, as expected. Also, the 100% test, which uses all data as both training and test data, it only gave an accuracy slightly higher than with 10% of training data. This might be due to, as previously said, this types of data having some of its rows being erratic and unpredictable (a patient that is punctual suddenly missing an appointment, or vice versa), which are extremely hard to predict. This is solved by the personalization section of the algorithm, which reacts to these types of behaviors, but the first shift in behavior is always hard to predict.

For the feature behavior tests, the health care center data was analyzed, in order to discover if there were any patterns, or any features that affected the patient's behavior more than others. The data consisted of around 608000 records and, for

Table 3: Age no-show values

Age	no-show probability
<18	28.3%
18-30	35.5%
30-65	36.7%
>65	32%

Table 4: Appointment date no-show values

Date	no-show probability
Weekday	34.7%
Weekend	43.9%

Table 6: Patient 1 profile personalization tests

Outcome	no-show probability
	37%
Attended	18%
Attended	12%
Attended	9%
Attended	7%
Attended	6%
Attended	5%
Missed	17%
Missed	26%
Attended	23%
Attended	21%

that data in particular, features that were found to affect more the patient's behavior is the patient's age and the date of the appointment (weekend or not). This can be seen in tables 3 and 4, which represent the no-show probabilities, only concerning those features in particular each. The other features (sex and distance) did not feature major patterns that were visible. All these results are available in the appendix.

After discovering these patterns, the logistic regression model was built with all the data and, for each different value for each feature, it was generated 10 different records, each with random values but with the feature in question consistent (10 random records with a male patient, 10 random records with an appointment on a weekend, and so on). These records were given to the logistic regression model and the average of each value for each feature was computed and compared to the "real" no-show probability for that feature. These "real" probabilities are not absolute, as they do not consider the effect of the other features, and are only shown as a means of comparison with the algorithm's results. The relevant results are represented in table 5.

From the obtained data, we can see that, for the most part, the algorithm behaves accordingly with the real data results. It was not expected that the output of the algorithm would match exactly as the real data values, as the algorithm is also taking into consideration the influence of all the other features, which will alter some of its results. Despite that, it is possible to conclude that the logistic regression model provides a good initial estimation of a patient's no-show probability based on the data that it's given, which will be further improved with the profile personalization section of the algorithm. One thing to note about this test in particular is that, as mentioned above, this test was done using the data of one health care center. While that gives a fair idea of the behavior of the general patient, it is still lacking and should be tested with multiple data sources in order to better detect the patterns and, most importantly, to determine the best features to use. Despite most of our research having cases of no-show solutions being applied to only one health care center, the algorithm would also only be applied to that specific health care center, while this algorithm will be applied to multiple across the country, and as such, that study is something to consider while looking at the results, and while making future test cases.

Table 5: Features no-show values

Feature	Value	Probability	Real
Age	<18	30.2%	28.3%
	18-30	39%	35.5%
	30-65	40.1%	36.7%
	>65	37.5%	32%
Date	Weekday	30%	34.7%
	Weekend	42.2%	43.9%

The last tests that were done to the health care center data were to test the profile personalization section of the algorithm and check how it compares to the patient's real behavior. For that, two different patients that had some kind of erratic behavior on their record were chosen. The logistic regression model was built without their appointments, and their first ten appointments were added to the model, one by one, in chronological model, and the resulting probability was compared to the actual outcome of the patient's decision. These results are represented in tables 6 and 7. The first row of each of those tables represent the initial probability for each of the patients.

One thing to note about the results is that, since human behavior is extremely complex, and there are lots of factors that can affect it, it is extremely hard to predict a sudden shift in behavior. A patient that has attended 5 appointments in a row is expected to keep up that behavior, and as such the algorithm will not be able to anticipate that initial shift. However, after it happens, as it is possible

Table 7: Patient 2 profile personalization tests

Outcome	no-show probability
	38%
Attended	19%
Missed	46%
Missed	59%
Attended	47%
Attended	39%
Attended	34%
Missed	42%
Attended	37%
Missed	43%
Attended	40%

to observe in patient 1 case, the algorithm immediately reacts, raising the no-show probability significantly, while also considering the previous attendance record.

Comparing both patients, it is possible to see that the second patient has only two extra missed appointments, but that represents a big difference on their no-show probability, making patient 1 preferable as a candidate replacement for the no-show algorithm, as that patient is ”punished” for missing the appointments, but rewarded again if they continue to attend appointments.

7. Conclusion

With this research we were able to conclude that the logistic regression model paired with the posterior mean would produce the expected results, being able to make good predictions of the general behavior of the population, while adapting that same probability accordingly for each patient. Through various scenarios, each testing different combinations of patient features and behaviors, the algorithm gave satisfying results, either for data created for those scenarios or for real world data from ”MD Clinica”. Also, the algorithm was able to deal with a big amount of data, which means that the model is able to be built in a reasonable amount of time, even if the number of information present on the database is extremely large. As this was done in a machine with a lower specs than the ones on which the web platform and the algorithm will run, the algorithm will perform even better on that environment.

During the development of this research, some difficulties and limitations were found. First of all, as we were researching the background for this research, we found that every implementation of a no-show management solution was used on only one health care center. The solutions also used these algorithms to estimate time slots where it would be most likely for a patient to miss an appointment, and as such it would allow the health care center to overbook that time slot in order to prevent that situation. On this case, our objective was to replace a patient, and as such we could not assume that the patient was not coming to the appointment, which meant that the no-show estimation could not be used to predict when and if a patient was going to miss the appointment, only for the candidate patients.

Another limitation, that could be improved in a continuation of this work, is the use of real world data to determine which features are the most appropriate for the Portuguese population and if those features change based on the patient’s location in the country (North, Center, South).

Acknowledgements

I would like to thank MedClick and Rui Cruz for the opportunity to be part of this project, and for all the support given during the whole year. Also, I would like to thank ”MD Clinica” for the data give that was very useful during the testing phase of the algorithm. Finally, I would like to thank my supervisor André Vasconcelos for the support during the making of this research.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining Association in Large Databases. *Proceedings of the 1993 ACM SIGMOD international conference on Management of data - SIGMOD ’93*, 1993.

[2] A. Alaeddini, K. Yang, P. Reeves, and C. K. Reddy. A hybrid prediction model for no-shows and cancellations of outpatient appointments. *IIE Transactions on Healthcare Systems Engineering*, 5, 2015.

[3] L. R. Bahi, P. F. Brown, P. V. D. Souza, and R. L. Mercer. Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition. 1986.

[4] H. Buhrman and R. De Wolf. Complexity measures and decision tree complexity: A survey. *Theoretical Computer Science*, 2002.

[5] J. Daggy, M. Lawley, D. Willis, D. Thayer, C. Suelzer, P.-C. DeLaurentis, A. Turkcan, S. Chakraborty, and L. Sands. Using no-show modeling to improve clinic performance. *Health Informatics Journal*, 16, 2010.

[6] E. Frank, M. Hall, L. Trigg, G. Holmes, and I. H. Witten. Data mining in bioinformatics using Weka. *Bioinformatics*, 2004.

[7] M. Fu and R. Storch. Reducing Disruptive Effects of Patient No-shows : A Scheduling Approach. 2013.

[8] P. Hall, B. U. Park, and R. J. Samworth. Choice of neighbor order in nearest-neighbor classification. *Annals of Statistics*, 2008.

[9] A. Krogh, È. Larsson, G. V. Heijne, and E. L. L. Sonnhammer. Predicting Transmembrane Protein Topology with a Hidden Markov Model : Application to Complete Genomes. *Journal of Molecular Biology*, 2001.

[10] W. L. W. Lee, S. J. Stolfo, and K. W. Mok. A data mining framework for building intrusion detection models. *Proceedings of the 1999 IEEE Symposium on Security and Privacy Cat No99CB36344*, 1999.

[11] S. Lemeshow and D. W. J. Hosmer. A Review of Goodness of Fit Statistics for Use in the Development of Logistic Regression Models. *American Journal Epidemiology*, 1982.

[12] T. Oates and D. Jensen. The Effects of Training Set Size on Decision Tree Complexity. *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.

[13] J. Ohya and N. T. T. H. I. Laboratories. Recognizing Human Act ion in Time-Sequent ial Images using Hidden Markov Model. 1992.

[14] Q. J. R. Decision trees and decision making. *IEEE trans. on systems, man and cybernetics*, 1990.

[15] E. L. L. Sonnhammer and A. Krogh. A hidden Markov model for predicting transmembrane helices in protein sequences. 1998.

[16] H. Tanaka, S. Uejima, and K. Asai. Linear Regression Analysis with Fuzzy Model. *IEEE Transactions on Systems, Man, and Cybernetics*, 1982.

[17] K. Wang, M. Li, D. Hadley, K. Wang, M. Bucan, K. Wang, M. Li, D. Hadley, R. Liu, J. Glessner, S. F. A. Grant, H. Hakonarson, and M. Bucan. PennCNV : An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data PennCNV : An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. 2008.

[18] O. R. Zaiane and J. Luo. Web usage mining for a better web-based learning environment. *Conference on Advanced Technology*, 2001.