Predicting Appointment Cancellations Using Patient Engagement Patterns (Medumo Group 6)

Jessica Sandler, Heather Johnson, Nicholas Pearce, Jacob Kozol
{sandlerj, heathdj, npearce, jkozol}@bu.edu

1. Project Task

Even with reminders, some patients are unable to attend their appointments. When patients cancel their appointments on short notice or do not show up for their procedure at all, it leaves vacancies in hospital schedules that could otherwise be used to serve other patients. The goal of our project is to help hospitals predict in advance which patients will not appear at their appointments, so they can use that space to accommodate other patients waiting for appointments.

The input is a dataset including a list of patients with certain demographic information about each patient. Each patient also has a set of interaction data with the medumo app, which was also used as part of the input. The output is a file including a column of user IDs and a column of 1's and 0's, with a 1 if the patient is predicted to miss their appointment or cancel within 3 days, and a 0 if the patient is predicted to attend.

This project provided a few challenges. First of all, the dataset was unbalanced, with only about 10% of the patients missing their appointments or canceling them on short notice. There was also some missing data, with about half of the patients missing age data and about a third missing gender data.

2. Related Work

[1] and [2] use, respectively, multinomial and binary logistic regression as the primary methods for solving the classification problem of predicting whether or not a patient will cancel/fail to appear at their appointment. [2] also uses L2-norm regularization to prevent overfitting, and 10-fold cross validation to assess the accuracy of their model. These studies led us to conclude that a logistic regression classifier would be a logical binary classification model to start with, and that cross validation would be an effective analysis tool.

3. Approach

Initially, we started with binary logistic regression because that is the method that many related works use. As our feature set grew, we decided to try other binary classifiers and compare their results to determine the best model.

We use binary logistic regression to separate the patients into the categories of took place (0) or missed appointment (1) with L2-norm regularization to counteract potential overfitting. We created this using Scikit Learn's logistic regression module to implement this using the newton conjugate gradient algorithm to optimize our solution. We gave it an input of 336 features.

$$|\mathbf{x}| = \sqrt{\sum_{k=1}^{n} |x_k|^2}$$

Equation 1: L2-Norm on the input vectors x

As our feature set grew we created a random forest classifier using Scikit learn as well. Random forests perform well on non-linear features and decision trees, in general, perform well even with irrelevant features. The random forest was trained on the same input as the logistic regression.

We also use four Support Vector Machines with a penalty parameter of 1. Because we do not know the exact shape of our data, we created four SVMs with linear, sigmoid, radial basis function, and polynomial kernels to accommodate for a variety of potential data configurations. We use the Scikit Learn SVC implementation because of its support for all of these kernels.

Additionally, we created a neural network using the Keras library, which runs Tensorflow in the backend. The input layer has an input dimension of 336 and an output dimension of 168 and uses a ReLU activation function on the weighted sum. The hidden layer applies ReLU again. The output layer has an output dimension of 1 and uses sigmoid activation to obtain a binary result. Weights are initialized using a uniform function and optimized with the Adam optimizer. Since the dependent variable is binary, the neural net uses the binary cross-entropy loss function. After trial and error, we decided to use a batch size of 330 and 200 epochs.

$$c = \sum_{0}^{1} p_i \log(1/q_i) = p_0 \log(1/q_0) + p_1 \log(1/q_1) = p_0 \log(1/q_0) + (1-p_0)\log(1/(1-q_0))$$

p_i is the actual probability and q_i is the predicted probability

Equation 2: Binary cross-entropy loss

Logistic regression works well if there is a clear decision boundary, whereas a random forest performs well on non-linear features and decision trees perform well even with irrelevant features. A logistic regression classifier has a lower likelihood of overfitting in comparison with a random forest, especially since we use L2-norm regularization. The neural network can handle a larger feature space than a logistic regression, so as we added features it began to outperform the logistic regression in analysis using only training data. However, because neural networks are more likely to overfit to a dataset, logistic regression performs better on the test set.

4. Dataset and Evaluation Metric

Table 1. Dataset summary

Positive	Negative	Ratio (pos/neg)
1763	234	7.53

Our dataset has 1997 training examples and 222 test examples. Each example includes an index, hospital ID, patient ID, registration date, procedure date, and whether or not the patient appeared at their appointment. We also have Medumo app engagement data for each patient, as well as some basic demographic and enrollment information for each patient (gender, date of birth, registration for email and SMS notifications). We use this data to create features for the month of the appointment date, the day of the week that the appointment falls on, the number of days between registration and appointment dates, the patient's age at the time of their appointment, the number of completed modules of each module type, and the number of messages received of each message type. In total, we had 336 features.

Cancellations within three days of colonoscopy appointment and no-shows are combined and labelled as 1. The procedures that took place are labelled as 0.

Patients' age and gender both have missing values at a rate of 46.9% and 29.8% respectively. We fill the missing values with the mean age and gender rather than dropping the entries altogether. Normalizing using the Standard Scalar means that replacing these missing values with the mean would have minimal effect on the data. We divided the data into 70 percent training and 30 percent test.

The training data was very imbalanced, with ~88.28% of the data points flagged as 0 and only ~11.72% of the data points flagged as 1. This meant that before we handled this problem, we were able to achieve high accuracies while obtaining low precision and recall. In order to handle this issue, we had to decide between oversampling the minority class (1) and undersampling the majority class (0). Because oversampling would have led to overfitting to the dataset, we have decided to accept the information loss associated with undersampling. Also, given the size our dataset the undersampled majority class will still be representative of the distribution for the full dataset.

Metric: The primary metric we use for analysis of the models is the area under the curve (AUC) of the receiver operating characteristic (ROC), which plots the true positive rate (TPR) against the false positive rate (FPR) of the binary classifications segmented at varying threshold levels, where TPR and FPR are defined as follows:

$$TPR = \frac{TP}{TP+FN}$$
$$FPR = \frac{FP}{FP+TN}$$
$$TP = True\,Positives$$
$$TN = True\,Negatives$$
$$FP = False\,Positives$$
$$FN = False\,Negatives$$

The shape of the ROC is produced by a trade-off between sensitivity, which is measured as the proportion of true positives correctly classified as such, and specificity, which is measured as the proportion of true negatives correctly classified as such. As the threshold used to segment prediction probabilities into binary classifications is lowered, more data points are classified as positive, which increases both the true positive and the false positive rates. Because TPR = sensitivity and FPR = 1 - specificity, an inverse relationship between sensitivity and specificity implies a direct relationship between TPR and FPR, as seen in the ROC, which is always monotonic.

The AUC of the ROC is equivalent to the probability that the model will assign a randomly chosen positive data point with a greater prediction probability than the prediction probability it assigns to a randomly chosen negative data point. The ROC produced by random classifications (AUC = 0.5) is the line from (0,0) to (1,1). ROCs with AUCs that take up more of the unit square are produced by better performing models.

We use this metric in conjunction with 10-fold cross-validation, which randomly splits the data into 10 folds and iteratively trains and tests models, holding out one fold as test data for each iteration. For each model, we compute the ROC and AUC for each of the 10 folds, then we compute the mean ROC and AUC across the 10 folds. Graphs displaying the results of this analysis can be seen in the Results section of this report.

The exclusive use of ROC AUC as a metric for performance initially led to an inflated sense of the robustness of the models. When the precision and recall of each model were analyzed, it was discovered that the high AUCs were the combined result of our models exclusively predicting the majority class (0) and the exceedingly high proportion of the majority class (~90%). To solve this issue, we balance the data by undersampling the majority class, as mentioned in our discussion of the data. This raised both precision and recall.

5. Results

Table 2: Kaggle Leaderboard as of 12/11/18, 11:30pm

#	change	Team	Score	Entries
1	new	Group 6 **(ours)**	0.87386	7
2	new	Colleen	0.58847	11
3	new	Lesley	0.58814	13
4	new	Sarah M'Saad	0.57077	36
5	-4	DTest	0.5	1

On the Kaggle leaderboard, we ran our logistic regression model on the 222 patient ids in the provided test file. Before balancing the data, our models predicted 0 for every sample.

Table 3: AUC Values using 10-Fold Validation:

Model	**Precision**	**Recall**	**Accuracy**	**Initial AUC**	**Final AUC**
Logit	0.92	0.85	0.85	0.74	0.92
Random Forest	1.00	1.00	0.99	0.67	0.83
Linear SVM	0.95	0.96	0.96	0.68	0.81
RBF SVM	0.92	0.92	0.92	0.50	0.50
Sigmoid SVM	0.78	0.88	0.88	0.50	0.50
Poly SVM	0.95	0.95	0.95	0.66	0.83
MLP	0.95	0.96	0.96	0.67	0.96

Because we were not given the results for the test set that was used to create our Kaggle submission, all of the above metrics were calculated using 10-fold validation on the training set.

Figure 1: Logistic Regression ROC Curve

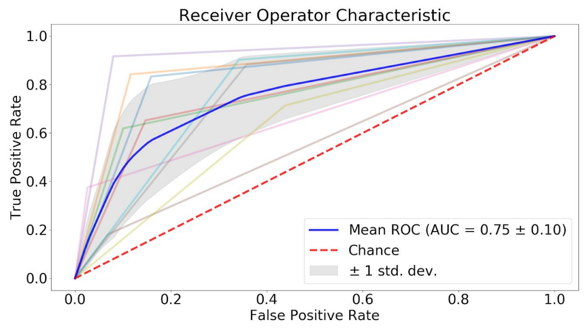


Figure 2: Random Forest ROC Curve

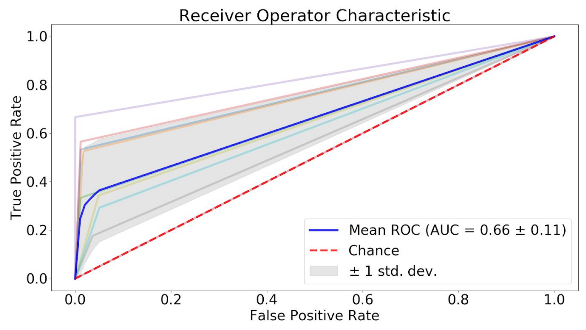


Figure 3: Neural Network ROC Curve

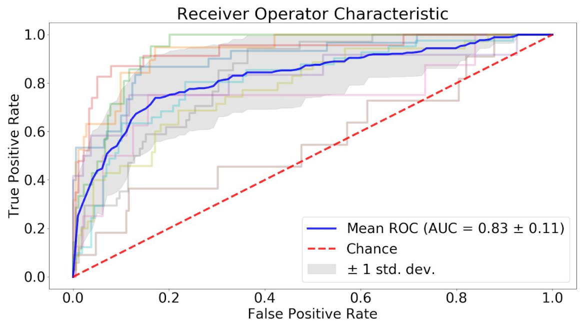
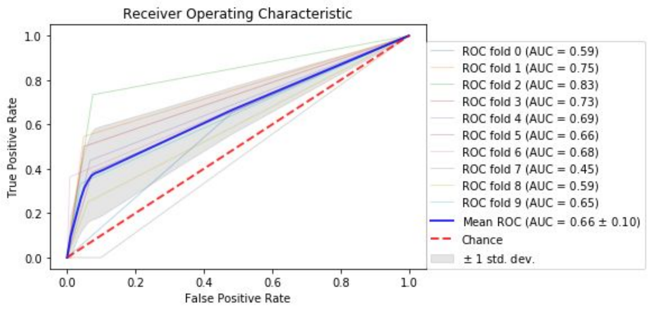


Figure 4: Polynomial SVM ROC Curve



Below are the confusion matrices for some of our models. They report the number of true positives and false positives in the first row and false negatives and true negatives in the second row, which provides a more reliable metric for classification given the imbalance of the dataset.

Table 4: Logistic Regression Confusion Matrix

1477	286
19	215

Table 5: Random Forest Confusion Matrix

1763	0
4	230

Table 6: Linear SVM Confusion Matrix

1751	12
174	60

Table 7: RBF SVM Confusion Matrix

1759	4
155	79

Table 8: Sigmoid SVM Confusion Matrix

1763	0
234	0

Table 9: Neural Network Confusion Matrix

1715	48
56	178

The best classifier is the logistic regression model. The neural network and random forest perform the best on the training data with high precision, recall, and accuracy. However, the logistic regression model produces a higher ROC AUC than the random forest, and it performs better than the neural network on the test data. The neural network overfits for the training data, so even though it performs very well on that data, it drops off on the test data. On the other hand, the logistic regression model avoids overfitting at the cost of accuracy and recall.

If we continued to work on this project, we would split up the data into more classes defined by the time of cancellation (3 days before, 7 days before, etc.). We would expand our neural network to classify for multiple classes instead of just two, and create a clustering model, which may be better suited to a problem with multiple classes.

Our predictions may also be improved with more data, as we only had about 2000 data points for training. This would prevent overfitting to our dataset and overall improve the performance of our models.

6. Timeline and Roles

Task	**Deadline**	**Lead**
Data Preparation	11/18/18	Nick
Implement Logistic Regression/ Random Forest	11/26/18	Jacob
Implement SVM	11/26/18	Heather
Implement Neural Network	11/26/18	Jessica
Validation/Analysis	throughout	Nick
Prepare Project Update I	11/15/18	all
Prepare Project Update II	11/29/18	all
Prepare report and presentation	12/11/18	all

References

1) *A. Alaeddini, K. Yang, P. Reeves, and C. Reddy. A hybrid prediction model for no-shows and cancellations of outpatient appointments. IIE Transactions on Healthcare Systems Engineering, 5(1):14-32, 2015.*
2) *H. Kurasawa, K. Hayashi, A. Fujino, K. Takasugi, T. Haga, K. Waki, T. Noguchi, and K. Ohe. Machine-Learning-Based Prediction of a Missed Scheduled Clinical Appointment by Patients With Diabetes. Journal of Diabetes Science and Technology, 10(3):730–736, 2016.*