Machine Learning Engineer Nanodegree

Capstone Project

Ihsaan Patel
June 13, 2017

I. Definition

Domain Background

Unplanned hospital readmissions, which occur when a patient is admitted into a hospital within a relatively short time period (e.g. 30 days) post discharge from a hospital, are a burden not only to patients but the entire healthcare system as well. One study examining Medicare readmissions estimated that unplanned readmissions represent a $12 billion burden on the U.S. healthcare system.[1] While interventions that could reduce readmission rates post discharge exist,[2] ensuring cost effective use of these interventions requires the ability to actually predict which patients are most at-risk for readmission. Studies attempting to predict readmission rates using machine learning have been done previously,[3] however advances in both natural language processing and machine learning algorithms since the time of those studies could lead to meaningful improvements in predictive capabilities.

Problem Statement

Given both administrative and clinical hospital data, is it possible to accurately predict a patient's risk of readmission at the time of discharge from the hospital? Readmission is defined as admittance into the same or another hospital facility within a 30 day time period. The problem is therefore a standard classification one (a patient was either readmitted within the defined time period or they were not) well-suited to machine learning techniques.

II. Data Exploration and Feature Engineering

Datasets and Inputs

The Medical Information Mart for Intensive Care (MIMIC) III database was the source of data for this project. MIMIC-III is a "large, freely-available database comprising deidentified health-related data associated with over forty thousand patients who stayed in critical care units

[1] http://www.nejm.org/doi/pdf/10.1056/NEJMsa0803563
[2] http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0109264
[3] http://content.healthaffairs.org/content/33/7/1123.abstract

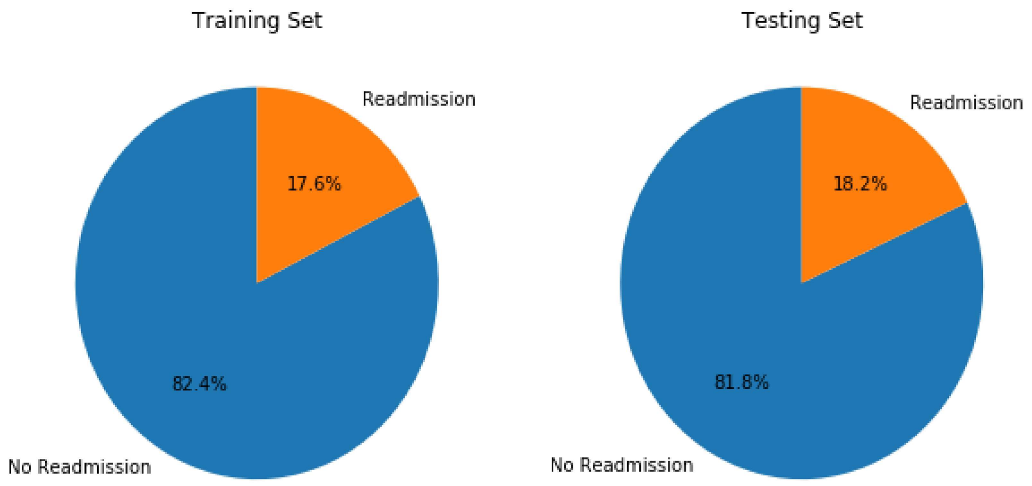
of the Beth Israel Deaconess Medical Center between 2001 and 2012." It includes a number of datasets containing demographic information, vital signs taken every hour while the patient was in the hospital, laboratory test results, procedures, medications, caregiver notes, imaging reports and mortality.[4]

Due to the nature of US regulations around patient data confidentiality and the anonymized nature of the MIMIC-III database, bringing in outside datasets was a non-starter. Therefore, the datasets in MIMIC-III were the only ones used in this project, however the large number of both datasets and records within the database still allowed for effective training and testing of machine learning models as discussed later in the paper.

Classifying Records and Splitting Data

The "admissions" dataset contained 58,976 records of all patient admissions, including multiple admissions for some patients. Only the first hospital admission was kept, and any patient with multiple admissions had this first admission classified as one with a readmission. Those admissions where the patient died in the hospital were also filtered out as a patient did not have a chance to be readmitted if they had passed away. After these transformations were complete, the dataset contained 42,115 records.

In preparation for model evaluation, the dataset was split into training and testing sets at 80% and 20% of the full dataset respectively. This left a substantial amount of records for training while still leaving enough testing records for accurate comparisons of model performance. As the charts below show, the proportion of classes was relatively equal between the training and test sets.



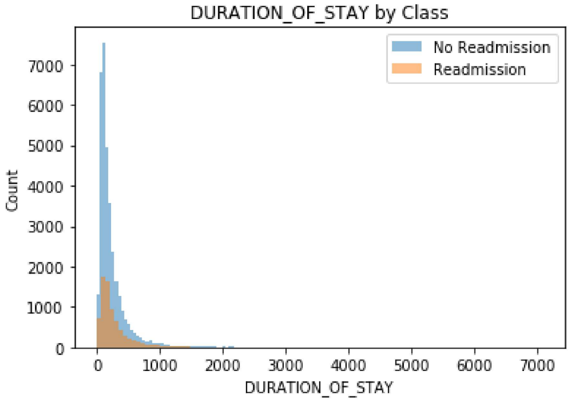
[4] https://mimic.physionet.org/

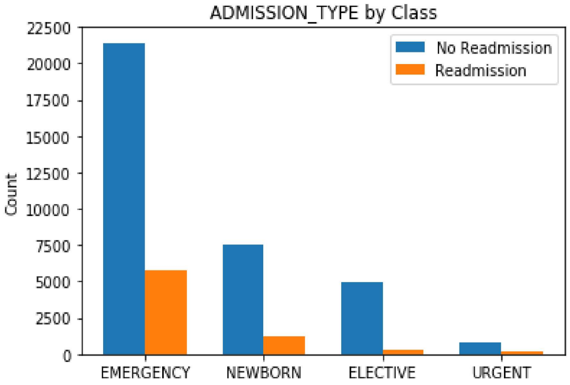
Exploratory Data Analysis and Preprocessing

Admissions Dataset

Duration of Stay (calculated as the difference between the admit and discharge times in hours) was chosen as a feature as readmitted patients appeared to have a similar amount of 40+ stays as non-readmitted patients, despite the significant difference in the total number of records between the two. Duration in the ER was chosen for a similar reason.

On the categorical side, Admission Type was also chosen given the differing proportions between ‘elective’ and ‘urgent’ admissions.





Other categorical variables chosen include admission type, discharge location, insurance, and ethnicity, with smaller categories consolidated into larger buckets to reduce dimensionality while still preserving some differentiation (i.e. Asian Indian and Korean were both consolidated into Asian). Language, religion,and marital status were not included in the feature set due to either a large amount of null values or large amounts of categories that could not be easily consolidated. All chosen categorical variables were encoded as dummy variables.

Current Procedural Terminology Codes Dataset

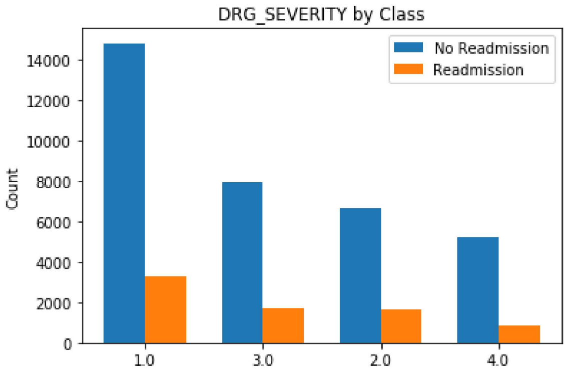
Each patient had one or many Cost Centers (ICU or Respiratory Unit), Section Headers (Medicine, Surgery, Radiology, etc.), and Subsection Headers (Hospital Inpatient Services, Pulmonary), and so each was first split into separate categories (with some consolidate on the Subsection side) and treated as a continuous feature.

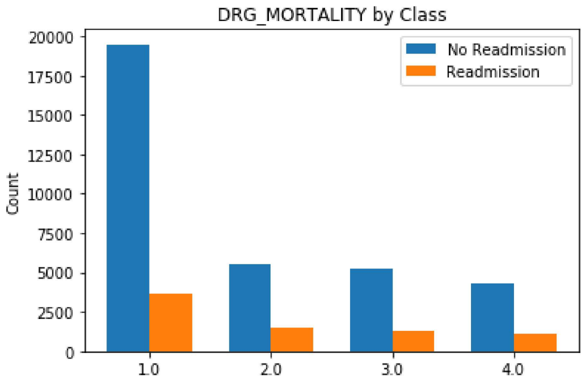
Chart Events Dataset

The number of chart events per patient was chosen as a continuous feature.

Diagnosis Related Group (DRG) Dataset

The DRG Severity, Mortality, and Description were chosen. Both Severity and Mortality were categorical variables that showed higher proportions of readmissions in the higher value categories, while Description was a text field which needed additional feature engineering (see text feature engineering section).



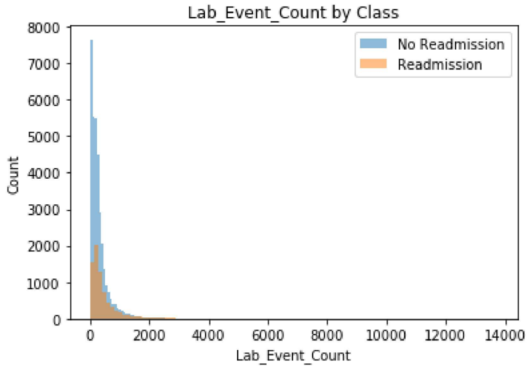


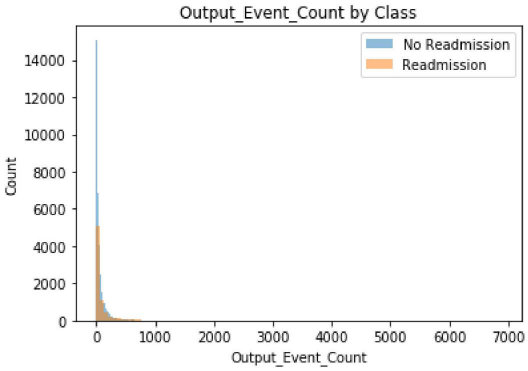
ICU Stays Dataset

The length of stay in the ICU was chosen as a continuous feature.

Input Events / Output Events / Lab Datasets

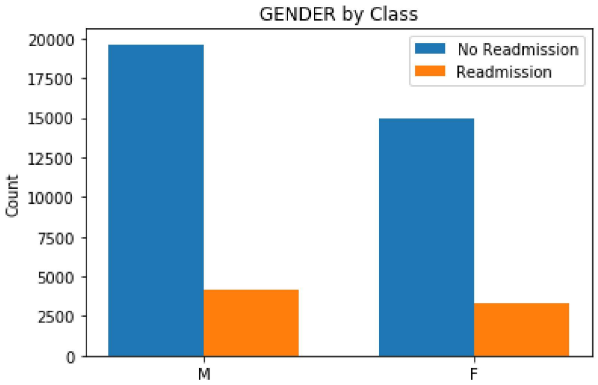
The number of input , output, lab and abnormal lab events were chosen as continuous features. Lab and output events seem to show significant variance between classes.

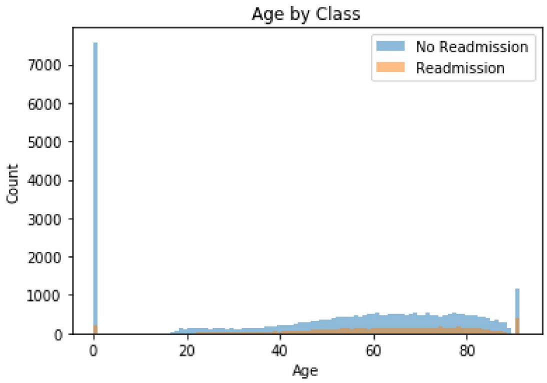




Patients Dataset

Gender and age (in years) were selected as features. Because birth dates had been shifted to preserve anonymity, age was calculated as the difference between the patient’s admission date and birth date. All patients older than 89 who were set to an age of 300 by MIMIC-III were set to their median age of 91.4. As the dataset shows, most patients in the database are newborns.





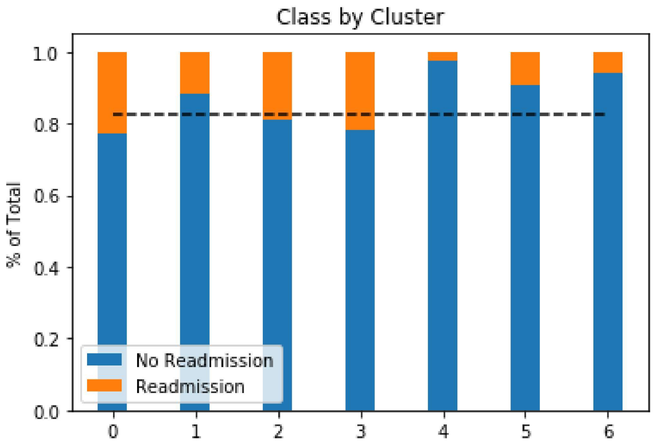
Services Dataset

The service that the patient was administered under was chosen as a categorical feature.

Text Feature Engineering

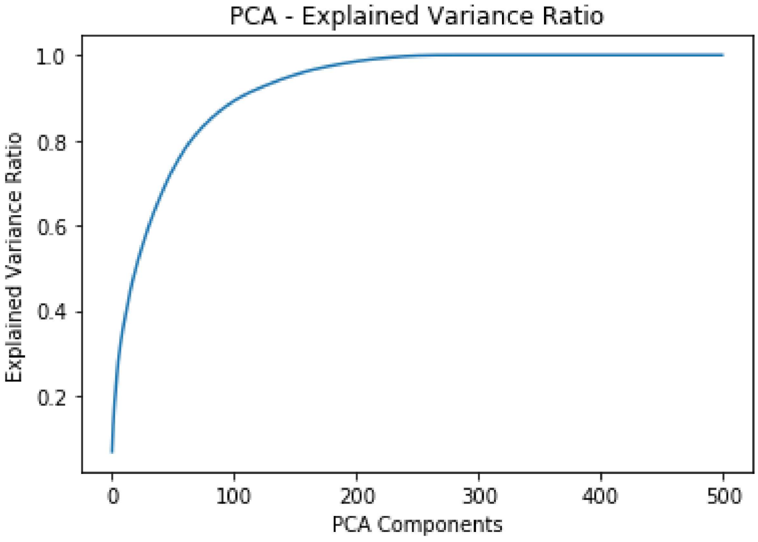
As discussed above, the DRG Description provided a potentially rich source of features but was a text field so it needed to be converted to features that could be used by machine learning models. To do this, each text field was first transformed into a term frequency - inverse document frequency (tf-idf) matrix which gets word counts for each record and then scales down the word counts based on the occurrence of the word in the entire training set. As part of the tf-idf transformation, words were “stemmed”, common english terms were filtered out, n-grams of lengths 1, 2, and 3 were used, the max dataset frequency was 15% (essentially the size of the readmission class), the minimum dataset frequency was 5 terms and the maximum amount of features was 100,000.[5]

To reduce the large dimensionality of the tf-idf matrix, both K-Means and Principal Components Analysis (PCA) were used. Both were fit on the training set only in order to prevent leakage from the test data. On the K-Means front, the number of clusters was tuned in order to create clusters where the proportion of classes was different than the proportion in training set. Eventually 7 clusters were selected, resulting in the following splits:



[5] Credit for some of the code and general approach goes to http://brandonrose.org/clustering

On the PCA side, the number of principal components was chosen by trying to balance the total explained variance with the number of components. Based on the results as displayed in the chart below, the top 20 components were chosen to be included as features in the dataset, explaining ~50% of the total variance in the tf-idf matrix:



III. Modeling Methodology

Benchmark Models

Three models were used as benchmarks for the project: a model that estimates readmission for no patients, a model that estimates readmission for all patients, and logistic regression model with l1 regularization. While outperforming the non-discriminative models was generally straightforward, the benchmark they provided may have been too low, and so the project models were also evaluated against the logistic regression model as this was the main machine learning model used in a previous study on the issue.[6] The LACE (length of stay, acuity of admission, Charlson comorbidity index, CCI, and number of emergency department visits in preceding 6 months) index, a simple model used in the medical community to predict risk of readmission, was also considered as a benchmark, however some of the data needed to calculate the index (especially number of emergency department visits in preceding 6 months) was not available in MIMIC-III. However, previous studies have shown relatively poor performance of the index, so outperforming the logistic regression model should be more than conservative as a proxy.[7]

[6] http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0109264
[7] https://www.hindawi.com/journals/bmri/2015/169870/

Evaluation Metrics

To compare the performance of the project's machine learning models with previous work done on the problem, the primary evaluation metrics were area under the ROC curve (AUC) and cost effectiveness. AUC provided a useful comparison on the trade-off between true and false positives, which represent costly mistakes if healthcare providers end up spending money on readmission prevention interventions that are not needed, and was used in the previous study.[8] Cost effectiveness was measured also based on the previous study, which calculated both a per patient cost for an unplanned readmission as well as for a readmission prevention program. The total cost of the model would be calculated as the sum of the following for each patient:

$$\text{Cost}_{\text{Intervention}} + \text{Probability}_{\text{Readmission}} \text{ x } \text{Cost}_{\text{Readmission}}$$

The cost effectiveness of the model was its total cost relative to the cost of either the all-intervention or no-intervention programs, depending on which one was lower. $\text{Cost}_{\text{Intervention}}$ was set at $1,300 and $\text{Cost}_{\text{Readmission}}$ was set at $13,679, both of which were taken from the aforementioned study.

Models and Tuning

Logistic regression with l1 regularization, random forest, gradient boosted trees, and were all used due to their widespread use in classification problems like the one in this project. SVMs, while initially considered, were ruled out from the options due to long training times and lack of probability outputs, which were needed for both AUC and cost effectiveness calculations. The sklearn package was used for the logistic regression and random forest models, while the xgboost package was used for the gradient boosted tree model.

All models' hyperparameters were tuned using grid search and 5-stratified fold cross-validation. The scoring metric for the grid search was actually the "F1" metric as it best approximated the balance between precision and recall needed to optimize cost effectiveness. AUC, precision, and recall were all insufficient alone to optimize this cost effectiveness given the imbalanced dataset.

IV. Mode Tuning and Performance

Logistic Regression

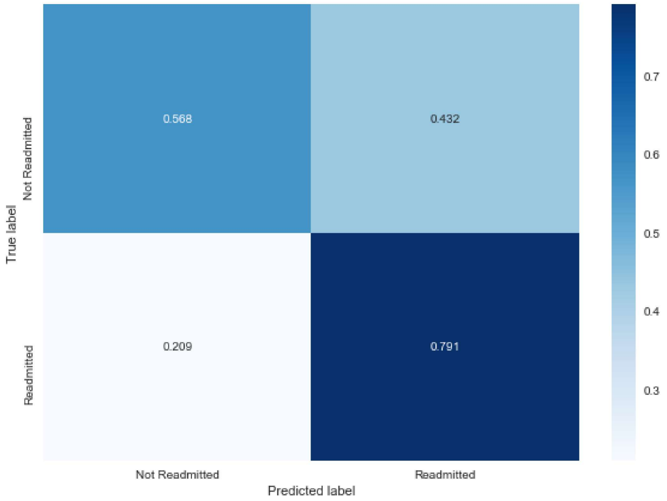
Tuned Parameters

Given the unbalanced nature of the dataset and the desire to use a benchmark similar to previous studies as discussed above, both the "penalty" and "class weight" parameters were fixed at "l1" and "balanced" respectively. The only parameter tuned was the magnitude of the penalty ("C") for which values between 10^{-5} and 10^{5} were tested. The optimal "C" parameter selected was 10^{1}.

[8] http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0109264

Performance

Despite the large number of features, the model performed similarly on both the training (AUC of 0.7414) and testing sets (AUC of 0.7423), indicating the l1 regularization did a good job of preventing overfitting. The test set confusion matrix however indicated that the model was too aggressive in predicting readmission, as more than 40% of non-readmissions were predicted to be readmissions.



Random Forest

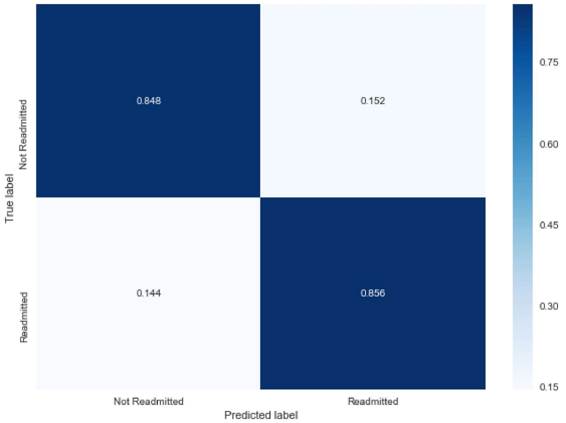
The random forest classifier grows decision trees while randomizing the subset of predictors used to grow those trees. The model then takes the mode of the outcomes of all the trees and records that class as the model's prediction.

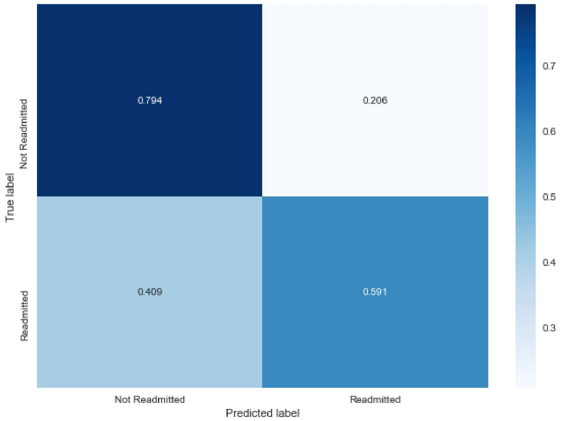
Tuned Parameters

As with logistic regression, “class weight” was fixed at “balanced”. In addition, the number of estimators (i.e. number of trees grown) was fixed at 1,000 as a balance between the amount of potential “learning” and computation time. The maximum number of features used to grow any individual tree was tuned on the following values: the square root of total features, the log2 of total features, 20% of all features, and 30% of all features. The minimum amount of samples per leaf, which could help control overfitting, was tuned at 1, 10, and 100. The maximum depth of the individual trees, which could also help prevent overfitting, was tuned at 10, 100, and None. Using all features, the chosen parameters were the maximum number of features at log2, the minimum samples per leaf at 10, and the maximum depth at 100.

Performance

The initial model tuned using all parameters appeared to significantly overfit the data as the training set had a significantly higher AUC (0.9300) compared to the testing set (0.7894) and the confusion matrices for each were quite different (training on the left, test on the right):





In order to test the hypothesis that some features were causing the model to overfit, I ran the model using only the top 10, 20, … , 90, 100 features by features importance. While the smaller feature models (particularly the top 10 model) appeared to overfit the training set less, performance in terms of AUC was still higher for the larger feature models, with the highest test set AUC coming from the top 100 model (0.7902).

XGBoost

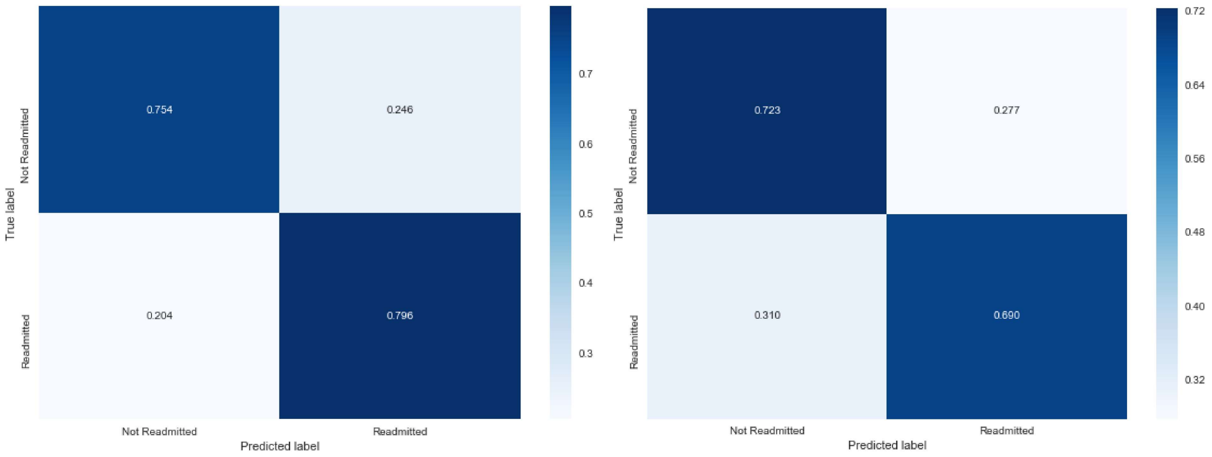
XGBoost is a GBM (generalized boosted model), and like random forest it uses decision trees. However, instead of completely randomizing the predictors in those trees, it learns from the residuals of previous trees and includes regularization. While this can lead to better model performance than random forest, it also makes the model more prone to overfitting.

Tuned Parameters

Similar to random forest., the number of estimators was fixed at 1,000 as was the “objective” at “binary:logistic” given the single class nature of the readmission problem. The learning rate, which controls how much information each tree learns from the residuals of the previous tree, was tuned at .001, .01, and .02. The minimum child weight, similar to minimum amount of samples per leaf in random forest, was tuned at 1, 5, and 10, while the number of column samples by tree, similar to max features in random forest, was tuned at 0.5 and 1. Gamma, which specifies the minimum amount of loss reduction needed to split the tree (and helps to control overfitting as well), was tuned at 0 and .1, and maximum depth was tuned at 3, 5, and 10. The initial model run appeared to underperform by improperly dealing with the imbalanced data (almost all records were predicted as non-readmission), and so another parameter, “scale_pos_weight”, was tuned at 5, 4, and 1. The chosen parameters for this balanced model were column samples by tree of 0.5, a “scale_pos_weight” of 4, a learning rate of 0.02, a minimum child weight of 5, a maximum depth of 5, and a gamma of 0.

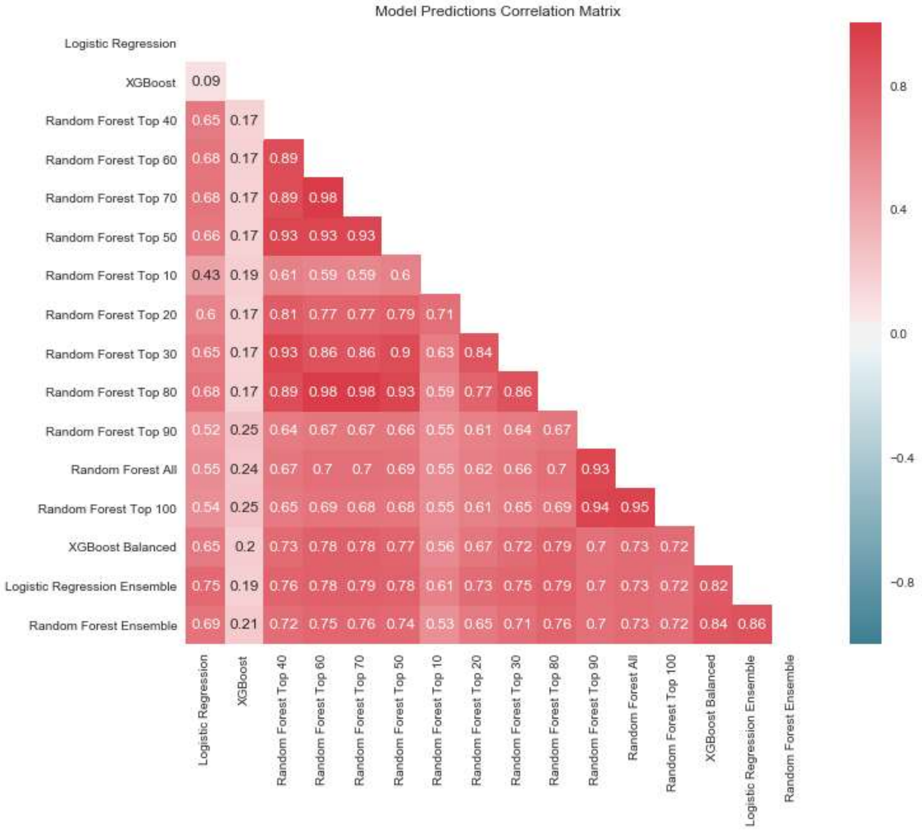
Performance

As discussed, the balanced XGBoost model performed significantly better than the non-balanced model (test set F1 score of .4707 vs. 0.2048). The balanced XGBoost model also maintained consistent performance between the training (AUC of 0.8650) and test (AUC of 0.7949) sets indicating good regularization (training on the left and testing on the right):



Ensemble

Ensembling of the individual models was also examined. Since the balanced XGBoost model performed the best, models whose predictions were less correlated with it (the unbalanced XGBoost, Logistic Regression, Random Forest Top 10, and Random Forest Top 90) were chosen to be included in the ensemble:


Model Predictions Correlation Matrix

	Logistic Regression	XGBoost	Random Forest Top 40	Random Forest Top 60	Random Forest Top 70	Random Forest Top 50	Random Forest Top 10	Random Forest Top 20	Random Forest Top 30	Random Forest Top 80	Random Forest Top 90	Random Forest All	Random Forest Top 100	XGBoost Balanced	Logistic Regression Ensemble	Random Forest Ensemble
Logistic Regression																
XGBoost	0.09															
Random Forest Top 40	0.65	0.17														
Random Forest Top 60	0.68	0.17	0.89													
Random Forest Top 70	0.68	0.17	0.89	0.98												
Random Forest Top 50	0.66	0.17	0.93	0.93	0.93											
Random Forest Top 10	0.43	0.19	0.61	0.59	0.59	0.6										
Random Forest Top 20	0.6	0.17	0.81	0.77	0.77	0.79	0.71									
Random Forest Top 30	0.65	0.17	0.93	0.86	0.86	0.9	0.63	0.84								
Random Forest Top 80	0.68	0.17	0.89	0.98	0.98	0.93	0.59	0.77	0.86							
Random Forest Top 90	0.52	0.25	0.64	0.67	0.67	0.66	0.55	0.61	0.64	0.67						
Random Forest All	0.55	0.24	0.67	0.7	0.7	0.69	0.55	0.62	0.66	0.7	0.93					
Random Forest Top 100	0.54	0.25	0.65	0.69	0.68	0.68	0.55	0.61	0.65	0.69	0.94	0.95				
XGBoost Balanced	0.65	0.2	0.73	0.78	0.78	0.77	0.56	0.67	0.72	0.79	0.7	0.73	0.72			
Logistic Regression Ensemble	0.75	0.19	0.76	0.78	0.79	0.78	0.61	0.73	0.75	0.79	0.7	0.73	0.72	0.82		
Random Forest Ensemble	0.69	0.21	0.72	0.75	0.76	0.74	0.53	0.65	0.71	0.76	0.7	0.73	0.72	0.84	0.86	

These models' cross-validation out-of-sample predictions were then added to the feature set, and then both a logistic regression model as well as a random forest model were fit and tuned (using the same parameters described earlier) on the feature set. Neither performed better than the best-performing individual models, perhaps due to the already high correlation between the various model predictions, and therefore were not included in the final model comparison.

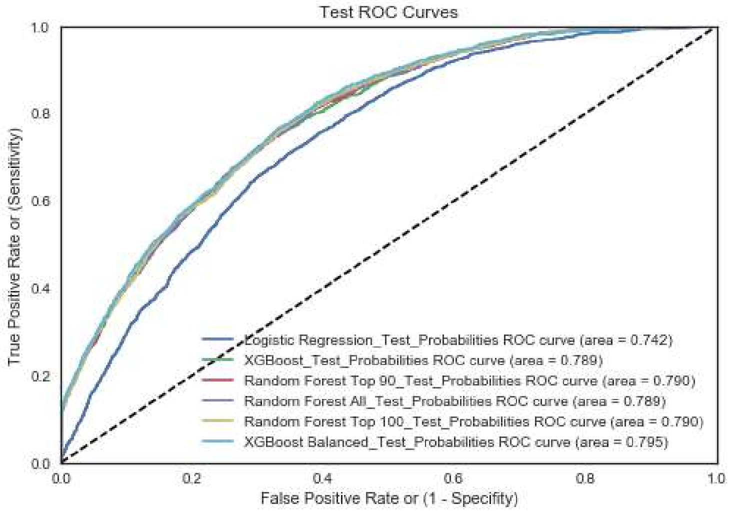
V. Model Results Comparison

AUC

As the table below indicates, the balanced XGBoost model performed the best in terms of AUC on the test dataset:

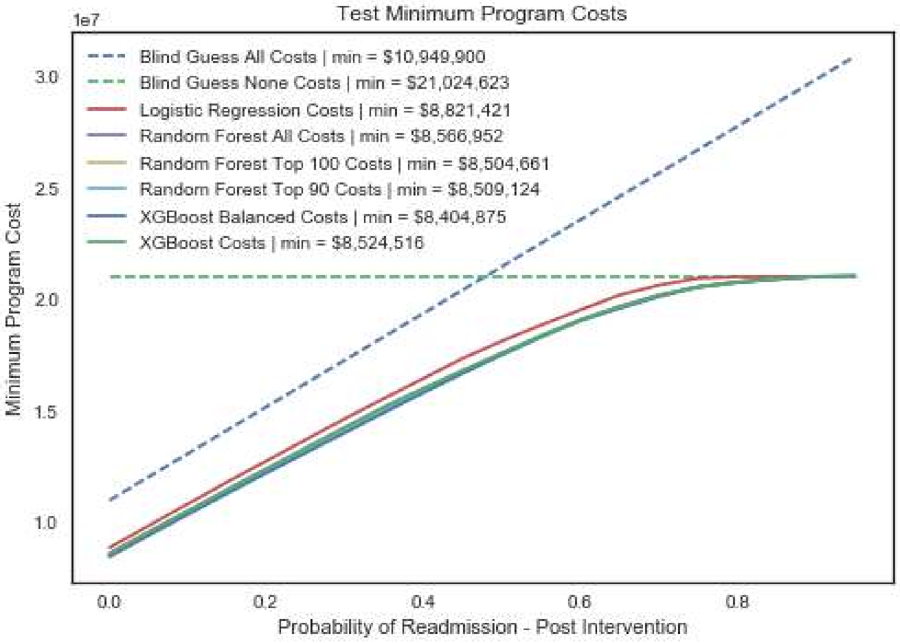
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**AUC**
XGBoost Balanced	0.716728	0.357119	0.690306	0.470719	0.794912
Random Forest Top 100	0.762080	0.396176	0.579701	0.470681	0.790185
Random Forest Top 90	0.767779	0.403233	0.567990	0.471637	0.789782
Random Forest All	0.756619	0.389867	0.590761	0.469736	0.789433
XGBoost	0.837706	0.967033	0.114509	0.204770	0.788757
Logistic Regression	0.608216	0.289769	0.790501	0.424084	0.742319
Blind Guess All	0.182477	0.182477	1.000000	0.308635	0.500000
Blind Guess None	0.817523	0.000000	0.000000	0.000000	0.500000

The performance of all the models was quite similar, however all of the tree-based models perform better than all of the benchmarks including the logistic regression used in the previous study, particularly at lower false positive rates:



Cost Effectiveness

The cost effectiveness of any interventional program that aimed to prevent readmissions by targeting high-risk patients would depend on the effectiveness of the intervention. It is unlikely that the interventional program would reduce the patient's probability of readmission to 0, however, so cost effectiveness was measured at different probabilities of readmission.



As the test set performance shows, all of the tree based models outperform the benchmarks at *all* probabilities of readmission and especially at higher ones, indicating that the interventional program would not need to be fully effective for the predictive model to have a cost saving impact. The best performing model, again the balanced XGBoost, would in a best case scenario save 23% compared to a program that intervened on all patients, 60% compared to a program that intervened on no patients, and 5% compared to a program that used the l1 logistic regression model to predict patient risk of readmission. In dollar terms, this could yield hundreds of millions if not billions of dollars in savings for the US healthcare system.

Feature Importance

Examining the feature importance of each model can help give insight into critical factors associated with predicting readmission. The table below indicates the top 10 features (in the case of logistic regression top 10 coefficients in terms of absolute value) for each model, with the color coding for logistic regression indicating a positive (green) or negative (red) coefficient. As the table indicates, text features appeared to play a vital role in the logistic regression model but are not as important in either of the tree-based models. While there is significant overlap between the random forest and xgboost models, there are some differences including the order of the different features. The most important features for both models appear to involve the duration of a patient's stay in the hospital and its various units, his or her age, and the number of interactions with the hospital (as measured by chart, input, output, and lab event counts).

Importance	Logistic Regression	Random Forest	XGBoost
1	Hospice-Medical Facility (Discharge Location)	Chart Count	Duration of Stay
2	Text PCA 0	Age	Age
3	Text PCA 2	Length of Stay - ICU	Abnormal Lab Count
4	Text Cluster 6	Abnormal Lab Count	Lab Event Count
5	Hospice-Home (Discharge Location)	Lab Event Count	Length of Stay - ICU
6	Text Cluster 5	Input Event Count	Chart Count
7	Text PCA 17	Output Event Count	Duration in ER
8	Text PCA 10	NB (Services)	Input Event Count
9	Text Cluster 0	Duration of Stay	Output Event Count
10	Text PCA 16	Admission Type (Discharge Location)	Critical Care Services (Subsection Header)

VI. Conclusion

The chosen model, XGBoost, outperformed all of the benchmarks in terms of both AUC and more importantly cost effectiveness. Implementation of the model could potentially lead to large cost savings for hospitals and health systems depending on the effectiveness of their interventional programs in reducing readmission risk. Even with this performance some potential improvements remain. Specifically, additional feature engineering could be done on the rich clinical data available in MIMIC-III, in particular the longitudinal data. In addition, deep learning models including multi-layer perceptrons and more ensembles could be applied to examine if there are any additional gains to be had.