Analysis of Machine Learning models to calculate survivability of ICU Patients with Severe Sepsis Shock

Sangeet Saurabh

Sep 28, 2017

Domain Background

Sepsis and severe sepsis (sepsis accompanied by acute organ dysfunction) are leading causes of death in the United States and the most common cause of death among critically ill patients in non-coronary intensive care units (ICU). Recent data suggest the annual cost of hospital care for patients with septicemia is $14 billion in United States. Also in the United States, the incidence of severe sepsis is estimated to be 300 cases per 100000 population. Approximately half of these cases occur outside the ICU. A fourth of patients who develop severe sepsis will die during their hospitalization. Septic shock is associated with the highest mortality, approaching 50%. The cumulative burden of organ failure is the strongest predictor of death, both in terms of the number of organs failing and the degree of organ dysfunction.Therefore, sepsis and severe sepsis are important public health problems.

Sepsis is a life-threatening condition that arises when the body's response to infection causes injury to its own tissues and organs. Most commonly, the infection is bacterial, but it may also be from fungi, viruses, or parasites. Common locations for the primary infection include lungs, brain, urinary tract, skin, and abdominal organs. Risk factors include young or old age, a weakened immune system from conditions such as cancer or diabetes, major trauma, or burns. Common signs and symptoms include fever, increased heart rate, increased breathing rate, and confusion. There also may be symptoms related to a specific infection, such as a cough with pneumonia, or painful urination with a kidney infection. In the very young, old, and people with a weakened immune system, there may be no symptoms of a specific infection and the body temperature may be low or normal, rather than high.Severe sepsis is sepsis causing poor organ function or insufficient blood flow. Insufficient blood flow may be evident by low blood pressure, high blood lactate, or low urine output. Septic shock is low blood pressure due to sepsis that does not improve after reasonable amounts of intravenous fluids are given.

Sepsis usually is treated with intravenous fluids and antibiotics. Typically, antibiotics are given as soon as possible. Often, ongoing care is performed in an intensive care unit. If fluid replacement is not enough to maintain blood pressure, medications that raise blood pressure may be used. Mechanical ventilation and dialysis may be needed to support the function of the lungs and kidneys, respectively. To guide treatment, a central venous catheter and an arterial catheter may be placed for access to the bloodstream. Other measurements such as cardiac output and superior vena cava oxygen saturation may be used. People with sepsis need preventive measures for deep vein

thrombosis, stress ulcers and pressure ulcers, unless other conditions prevent such interventions. Some might benefit from tight control of blood sugar levels with insulin.

Problem Statement

Although U.S. hospitals pay out over $22.2 billion annually for sepsis treatment, the disease still causes 20-45% of death in hospitals. By the time the physical manifestations of sepsis start to appear, it's sometimes too late to help the patient. Somewhere, buried in the data – in the blood values, the blood-pressure values, the heart rate, the temperature – is a prediction that this patient is heading toward an overwhelming infection and is likely to die. If we could predict that very early, it will make it less likely the patient would die.

A predictive model that determines which patients are most vulnerable can really help ICUs focus their Physicians and staff on the most vulnerable patients. Potentially, such selective treatment can reduce mortality without increasing spending. This machine learning model is being designed to predict death within 30 days of ICU admission for the patients who have been diagnosed with Sepsis/septic shock.

Metrics

Based upon mimic-iii database, we know the patients who were diagnosed with Sepsis shock and didn't survive within 30 days of admission. Using that data, the goal is to predict the sepsis shock patients who are not likely to survive beyond 30 days of admission. This information will provide ICU physicians and staff right set of patients to focus their time, effort and resources on for most effectiveness.

From the goal, it appears that using **accuracy** as a metric for evaluating a particular model's performance would be appropriate. But if model makes a mistake in identifying patients who will survive as someone who won't survive, it will be a sheer waste of ICU physicians and staff's time, effort and resources. Therefore, a model's ability to precisely predict those who won't survive is more important than the model's ability to **recall** those patients. As a result, **F-beta score** as a metric that considers both **precision** and **recall** will be more important for this model. We will use F-beta score as a metric that considers both precision and recall -

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

In particular, when beta=0.5, more emphasis is placed on precision. This is called the F score (or F-score for simplicity).

Looking at the distribution of data (those who survive past 30 days, and those who don't survive), it's clear that most patients survive (~80%) past 30 days. This can greatly affect **accuracy**, since we could simply say *"this patient survives past 30 days"* and generally be right, without ever looking at the data! Making such a statement is called **naive**, since we have not considered any information to substantiate the claim. As we build the machine learning model, we will look at Naive prediction and make sure that our Machine learning model performs better than Naive prediction. Here is the Naive prediction for the patient not surviving past 30 days -

Accuracy: 21.3922%

F-score: 0.2538

Solution Approach

The solution to is to categorize patients survivability at the time of admittance. This is calculated by considering: patient's demographic, vitals and infection information. A supervised learning model will be trained off of this information along with the calculated target value, which is the patient survival over 30 days from admittance.

The general strategy to solve this problem is to build a machine learning model using right set of features available within MIMIC-III database. Here are the steps that are taken to solve the problem -

1. From MIMIC-III database, all the patients who have been diagnosed with Sepsis are filtered out.

2. Sepsis diagnosis customers will be categorized into 2 categories - patients who died within 30 days of admission and patients who survived past 30 days.

3. Following categories of features are analyzed to figure out the right set of features to build machine learning model -
    a. Patient's age, gender, insurance information etc.
    b. Patients vitals collected within 48 hours of admission
    c. Patients infection that's reported within 24 hours of admission

4. Once features are finalized and all the data are available, multiple models using different Machine learning algorithm are implemented, analyzed and a comparative analysis between them are done. Based upon the comparative analysis, results are published.

Data Exploration and Visualization

For this study, the MIMIC-III dataset(https://mimic.physionet.org/) will be used. This dataset is accessible after taking small "Data or Specimens Only Research" course offered by MIT and suggested for use by Udacity. More than 40,000 individual patient entries will be utilized. 1184 patients were admitted with sepsis and thousands more diagnosed by the end of their stay. Focus will be put on the patients admitted with sepsis since there is an associated time of admittance that will aid in finding more effective treatments. Diagnosis data is tied to ICD-9 billing codes that do not have a time associated.

The data was downloaded locally as CSV files. The provided conversion scripts were modified to load the entries into the remote Amazon RDS Postgres database. This database and server is secured and is HIPAA compliant. An Amazon compute EC2 instance has been used to do most of machine learning related heavy lifting.

Demographic data, lab results and bacterial infections data of Sepsis shock patients were used as input features. Here are the list of tables from the database that were used to extract data to build machine learning models:

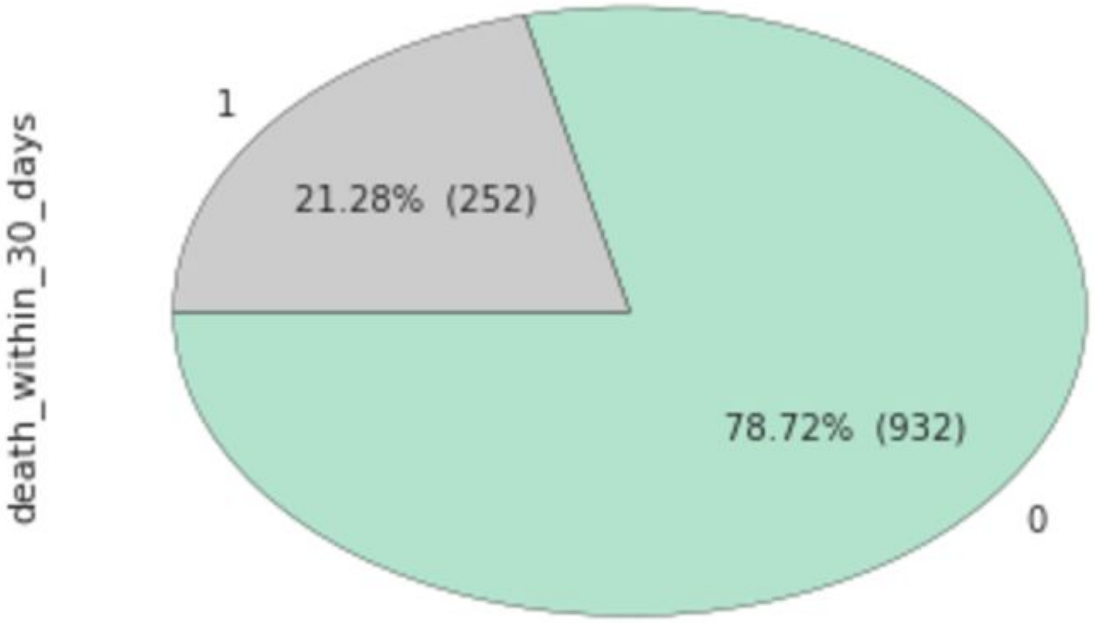
Admissions

The ADMISSIONS table gives information regarding a patient's admission to the hospital. Information available includes timing information for admission and discharge, demographic information, the source of the admission, and so on. More detailed information available here - https://mimic.physionet.org/mimictables/admissions/ .

Here are the steps that were taken to prepare Admissions Data for machine learning model -

1. **Retrieve Admissions data for all Sepsis Shock patients -** Using sqlalchemy, all patients and admission information were retrieved.

2. **Calculate the target label for Supervised learning -** Since our machine learning model is a supervised learning classification problem, the target is calculated using the information available in the Admission table. The goal is to predict survivability within 30 days of admittance. That data is not available by default. But the admissions table has a date of death (dod) as well as date of admission to ICU. Using these 2 fields, survivability within 30 days of admittance is calculated and used as target. Here is the distribution of data across the patients who survive vs patients who didn't survive -



**1 - indicates the patient who died within 30 days of admission (i.e. 21.28%)

**0 - indicates the patient who survived (78.72%)

3. **Drop data that's not needed for machine learning model** - Not all data in Admissions table to run machine learning models. So, following columns are dropped as they will have no impact on survivability -

1. dischtime
2. deathtime
3. admission_location
4. discharce_location
5. language - ethinicity is included if there is a variation based upon origin/race of the person.
6. religion - ethinicity is included if there is a variation based upon origin/race of the person.
7. marital_status
8. edregtime
9. edouttime
10. diagnosis (only sepsis customers are there in this DB)
11. hospital_expire_flag

After dropping not needed columns, table looks like -

	subject_id	hadm_id	admittime	admission_type	insurance	ethnicity	death_within_30_days
0	357	122609	2198-11-01 22:36:00	EMERGENCY	Private	WHITE	0
1	366	134462	2164-11-18 20:27:00	EMERGENCY	Medicare	HISPANIC OR LATINO	0
2	94	183686	2176-02-25 16:49:00	EMERGENCY	Medicare	ASIAN	0
3	21	111970	2135-01-30 20:50:00	EMERGENCY	Medicare	WHITE	1
4	353	108923	2151-03-28 16:01:00	EMERGENCY	Medicare	WHITE	0

Patients

Contains all charted data for all patients. This provides basic information about the patient including date of birth, date of death (if applicable), gender, etc. More detailed information available at - https://mimic.physionet.org/mimictables/patients/ .

Here is how Patients data is prepared and made ready for Machine learning models -

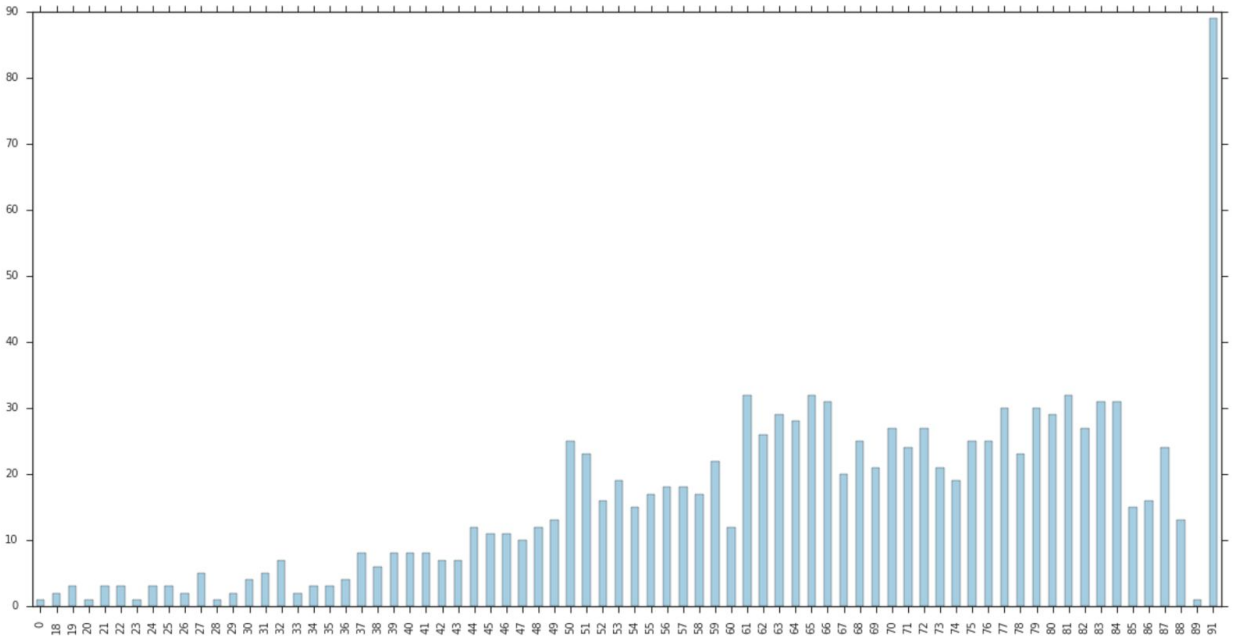
1. **Join Patients and Admissions Data** - After picking up Date of Birth and Gender information from Patients table, data is joined to Admissions Data. Integrated data looks like -

	subject_id	hadm_id	admittime	admission_type	insurance	ethnicity	death_within_30_days	gender	dob
0	357	122609	2198-11-01 22:36:00	EMERGENCY	Private	WHITE	0	M	2135-03-22
1	366	134462	2164-11-18 20:27:00	EMERGENCY	Medicare	HISPANIC OR LATINO	0	M	2112-05-22
2	94	183686	2176-02-25 16:49:00	EMERGENCY	Medicare	ASIAN	0	M	2101-09-20
3	21	111970	2135-01-30 20:50:00	EMERGENCY	Medicare	WHITE	1	M	2047-04-04
4	353	108923	2151-03-28 16:01:00	EMERGENCY	Medicare	WHITE	0	M	2089-07-23

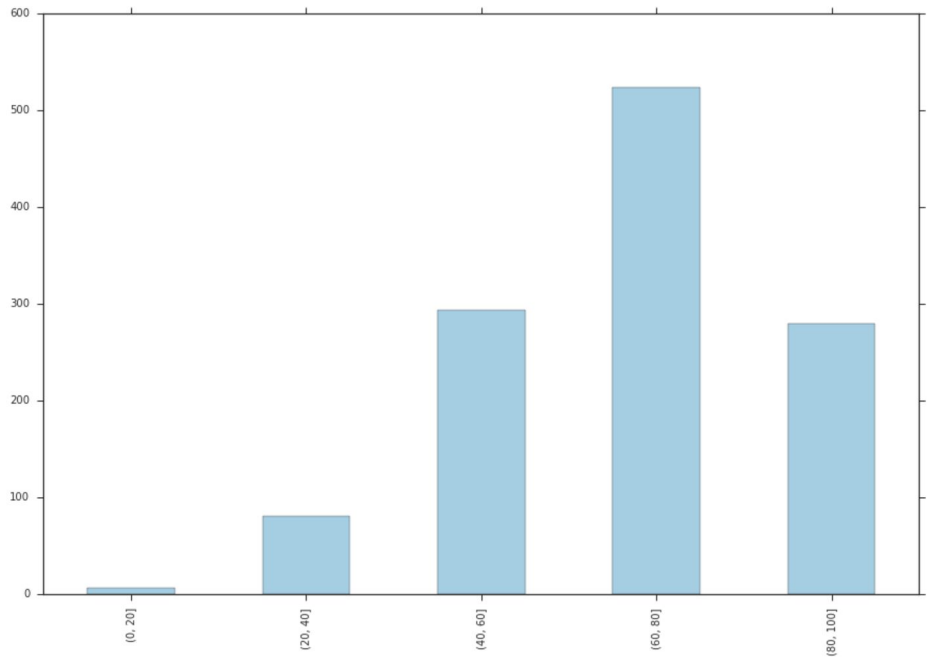
2. **Calculate patient's age at the time of admission** - Using date of birth and Admit date, patient's age is calculated at the time of admission. **Note:** Patients who are older than 89 years old at any time in the database have had their date of birth shifted to obscure their age and comply with HIPAA. The shift process was as follows: the patient's age at their first admission was determined. The date of birth was then set to exactly 300 years before their first admission. Since median age of such patient was 91.4, I am setting up age of such patient as 91.

3. **Analyze the data** - Once Admissions and Patients data are integrated together, I analyzed the data to better understand the data. Here are some of the visual analysis I did -

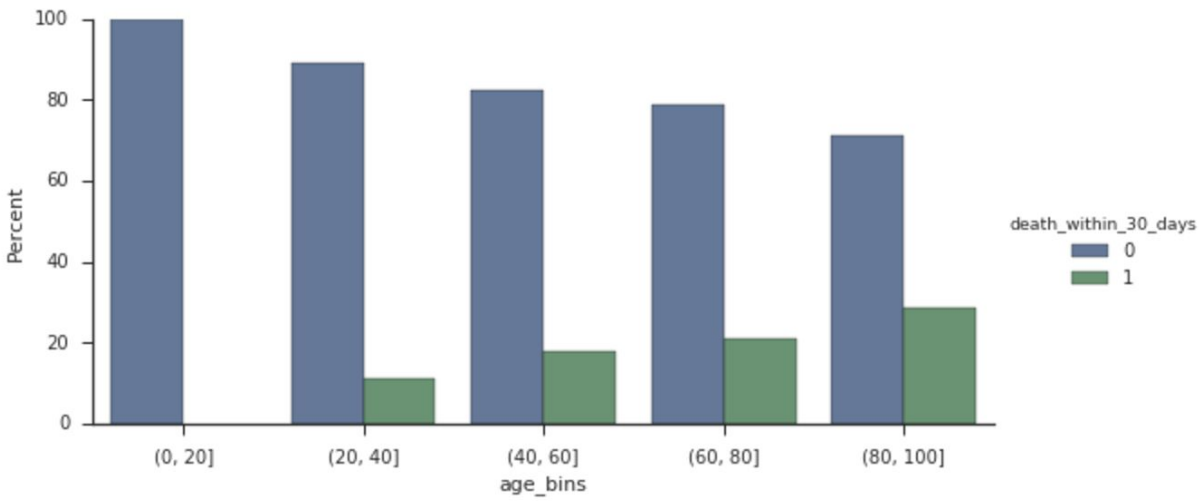
- **Number of Patients by Age -**



- **Number of patients by age groups** - Patients were placed into different age groups - 0 to 20, 20 to 40, 40 to 60, 60 to 80, and 80 to 100.



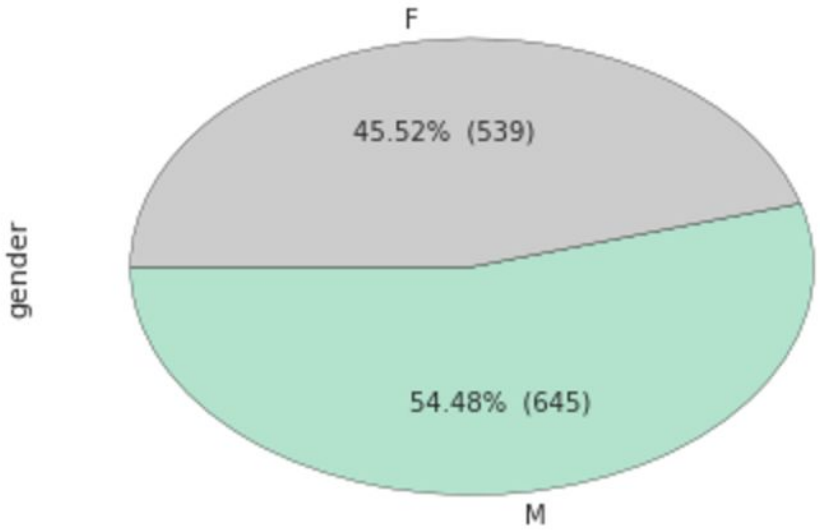
- **Survivability of patients by different age groups** -



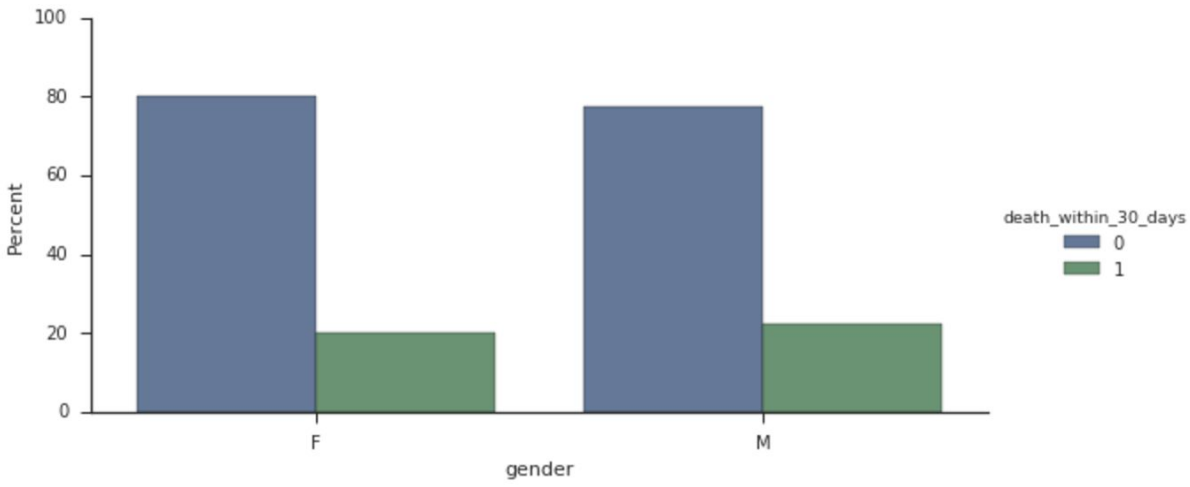
** 1 - indicates the patient who died within 30 days of admission

** 0 - indicates the patient who survived

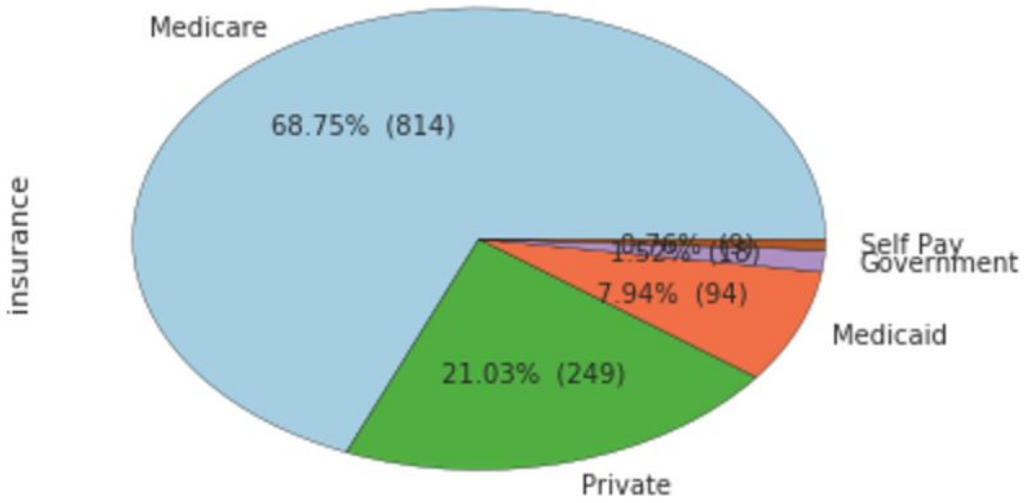
- **Patients by Gender** -



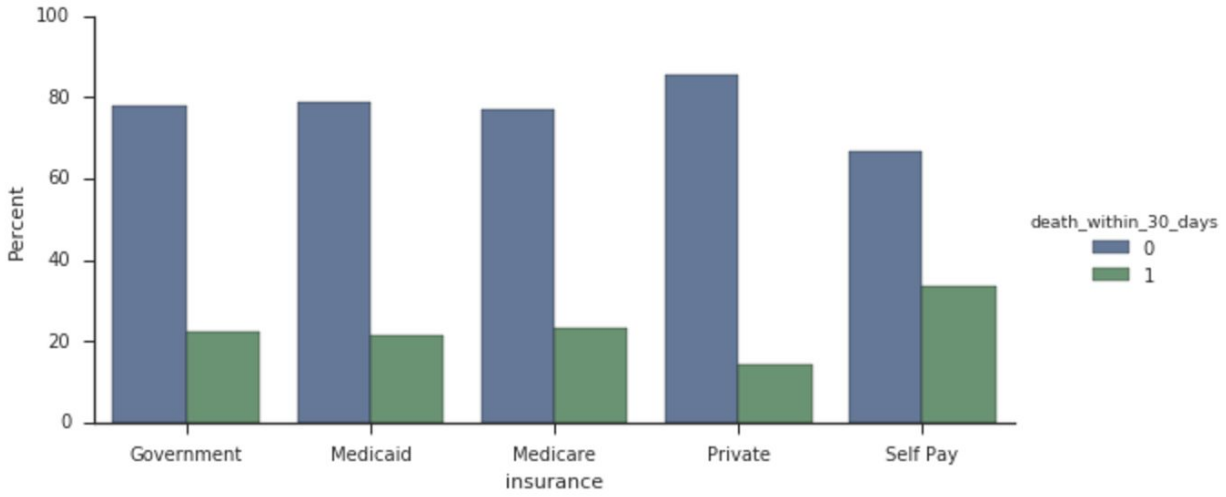
❑ **Survival and Death by percentage across both genders -**



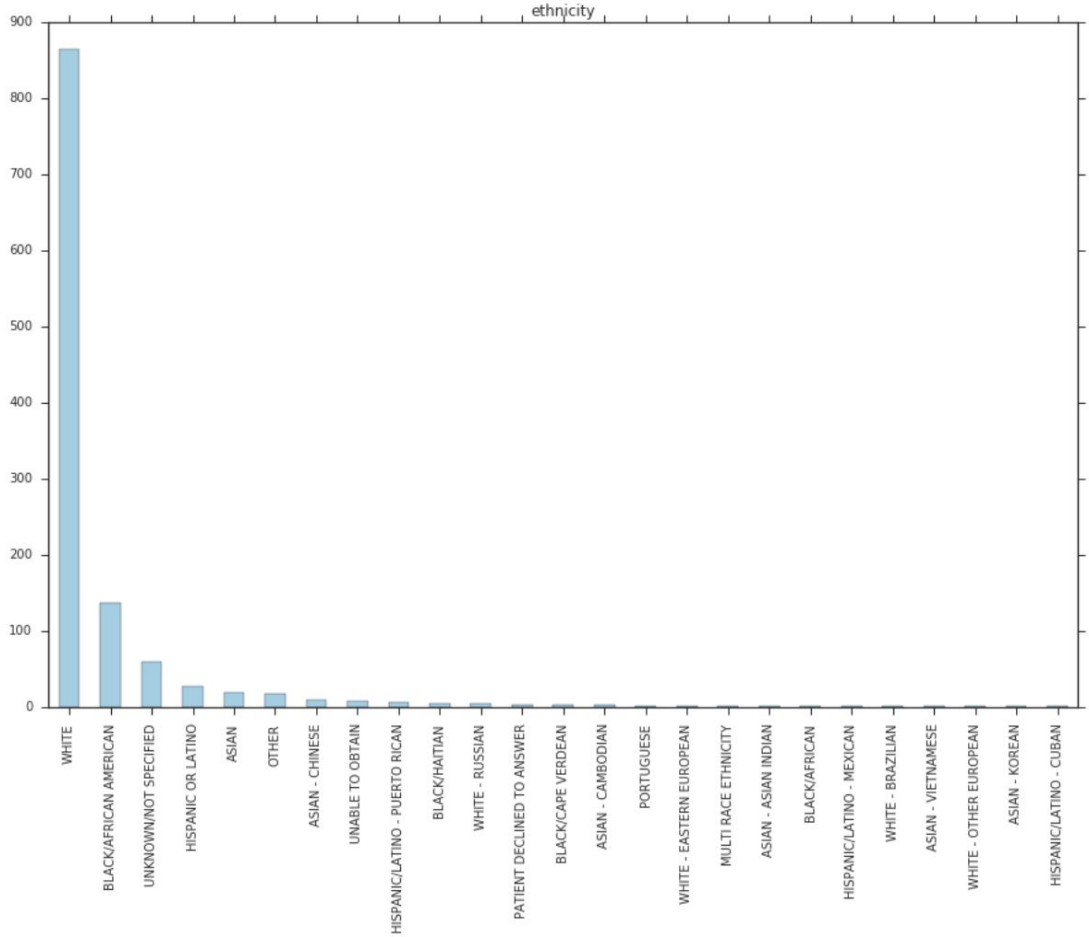
❑ **Patients by insurance -**



❑ **Survivability of Patients by insurance** -



- **Patients by Ethnicity -**



Labevents

The LABEVENTS data contains information regarding laboratory based measurements. The process for acquiring a lab measurement is as follows: first, a member of the clinical staff acquires a blood from a site in the patient's body. Next, the blood is bar coded to associate it with the patient and timestamped to record the time of the fluid acquisition. The lab analyses the data and returns a result within 4-12 hours. These lab tests are used as features for machine learning models that we are going to build.

Here are the steps taken to prepare vital data for machine learning model -

1. **Join Vital (labevents) data and Patient Admission data** - Patients vital and admission data are joined together.

2. **Filter out data that's captured within 24 hours of admission** - Note that we are predicting the patient's survivability within 24 hours of admission. So, we will use only the data that was captured before 24 hours of hospital admission.

3. **Retrieving Lab Item label from Lab events table** - D_LABITEMS contains definitions for all ITEMID associated with lab measurements in the MIMIC database. All data in LABEVENTS link to the D_LABITEMS table. Each unique LABEL in the hospital database was assigned an ITEMID in this table, and the use of this ITEMID facilitates efficient storage and querying of the data.

4. **Figure out which Vitals to use for Machine Learning Model** - Note that data is being captured for 409 vital event type (or lab items type). Now, we will analyze this data to figure out which ones to use for machine learning model. We will use the ones that are being

consistently captured across all the patients. Here is the logic - We have total of 1184 hospital admissions. If a lab item (or vital) data exists for at least 80% of the hospital admission (i.e. 948 hospital admissions), then we are going to use that lab item to build our machine learning model. Based upon this logic, 36 lab items were picked up for machine learning models. Here are the lab items that were picked up -

itemid	label	fluid
50813	Lactate	Blood
50861	Alanine Aminotransferase (ALT)	Blood
50863	Alkaline Phosphatase	Blood
50868	Anion Gap	Blood
50878	Asparate Aminotransferase (AST)	Blood
50882	Bicarbonate	Blood
50885	Bilirubin, Total	Blood
50893	Calcium, Total	Blood
50902	Chloride	Blood
50912	Creatinine	Blood
50931	Glucose	Blood
50960	Magnesium	Blood
50970	Phosphate	Blood
50971	Potassium	Blood
50983	Sodium	Blood
51006	Urea Nitrogen	Blood
51146	Basophils	Blood
51200	Eosinophils	Blood
51221	Hematocrit	Blood
51222	Hemoglobin	Blood
51237	INR(PT)	Blood
51244	Lymphocytes	Blood
51248	MCH	Blood
51249	MCHC	Blood
51250	MCV	Blood
51254	Monocytes	Blood
51256	Neutrophils	Blood
51265	Platelet Count	Blood
51274	PT	Blood
51275	PTT	Blood
51277	RDW	Blood
51279	Red Blood Cells	Blood
51301	White Blood Cells	Blood
51491	pH	Urine
51492	Protein	Urine
51498	Specific Gravity	Urine

For some lab items, data has been captured but valid values haven't been entered. Value entered is null. For lab items, that has too many invalid entries are discarded.

microbiologyevents

This table provides information about whether or not an infection is present, how it was obtained, and when.Note The MICROBIOLOGYEVENTS table does not contain cultures from samples taken outside the ICU. This table provides data on infections that are used as features for machine learning model.

Here are the steps taken to prepare infection information for machine learning model -

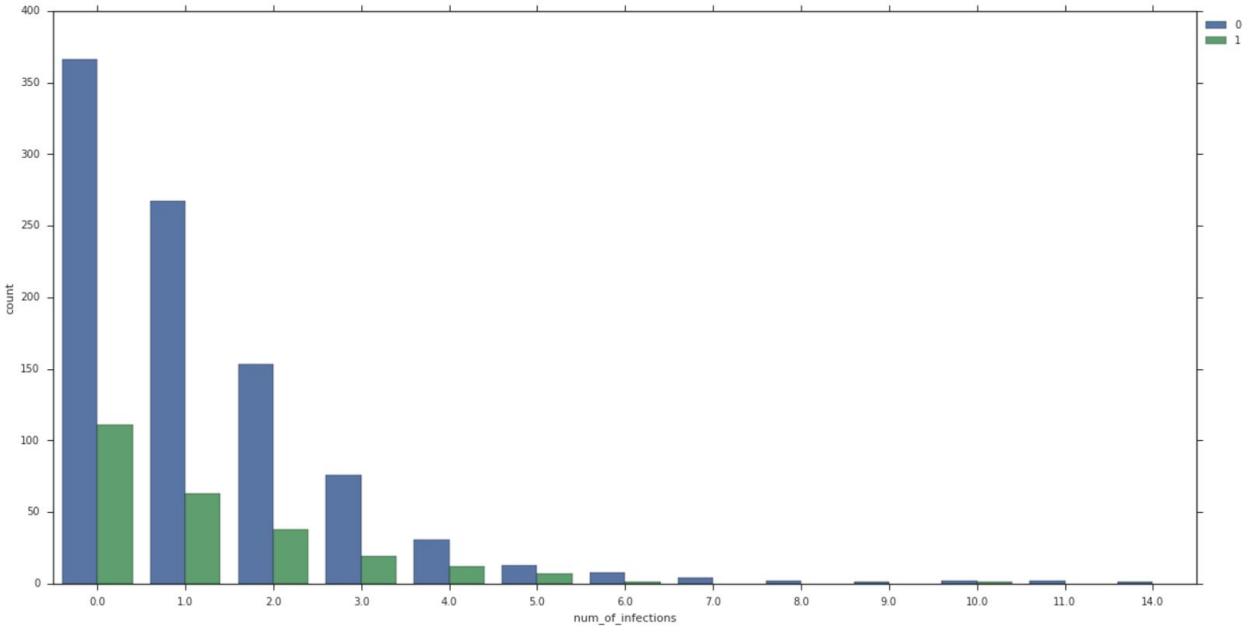
- **Retrieve Micro Biology events** - For all the patients above, micro biology events were retrieved from MICROBIOLOGYEVENTS table.

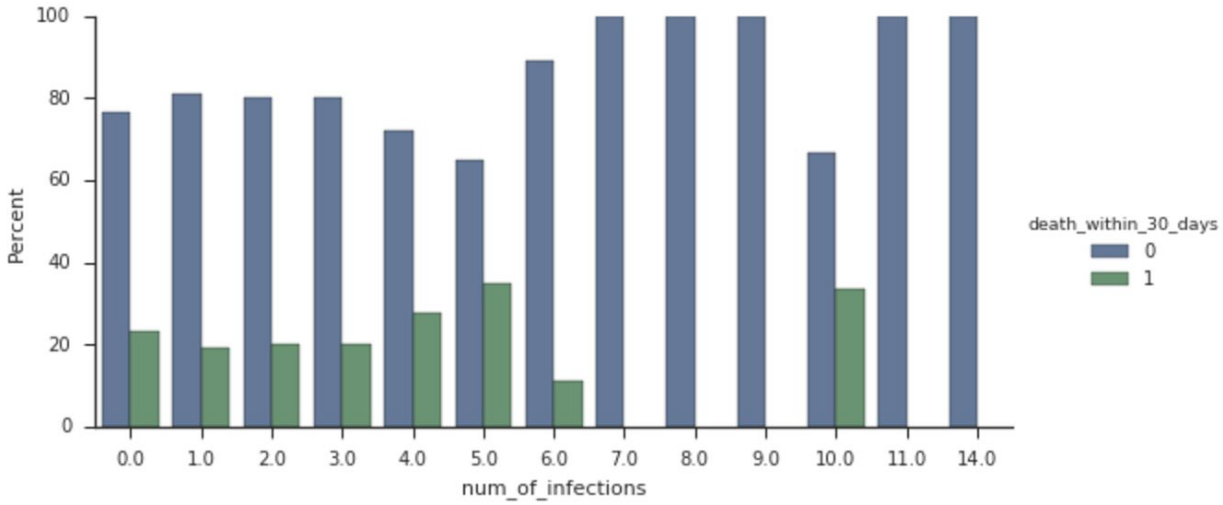
- **Filter out options reported within 24 hours of admission** - Since survivability is being predicted based upon first 24 hours of data, that data is being filtered out. Also, picking up data 48 hours prior as many times admit time is recorded late or admission happens after all the testing.

- **Figure out which features to use for machine learning models** - First goal was to figure out which dataset to use to build machine learning model. Following options were considered -
  - Option 1 - Number of infections a patient has
  - Option 2 - Infection by Specimen type (e.g. Sputum infection, Blood infection etc.)
  - Option 3 - Organism that's present in patient's body

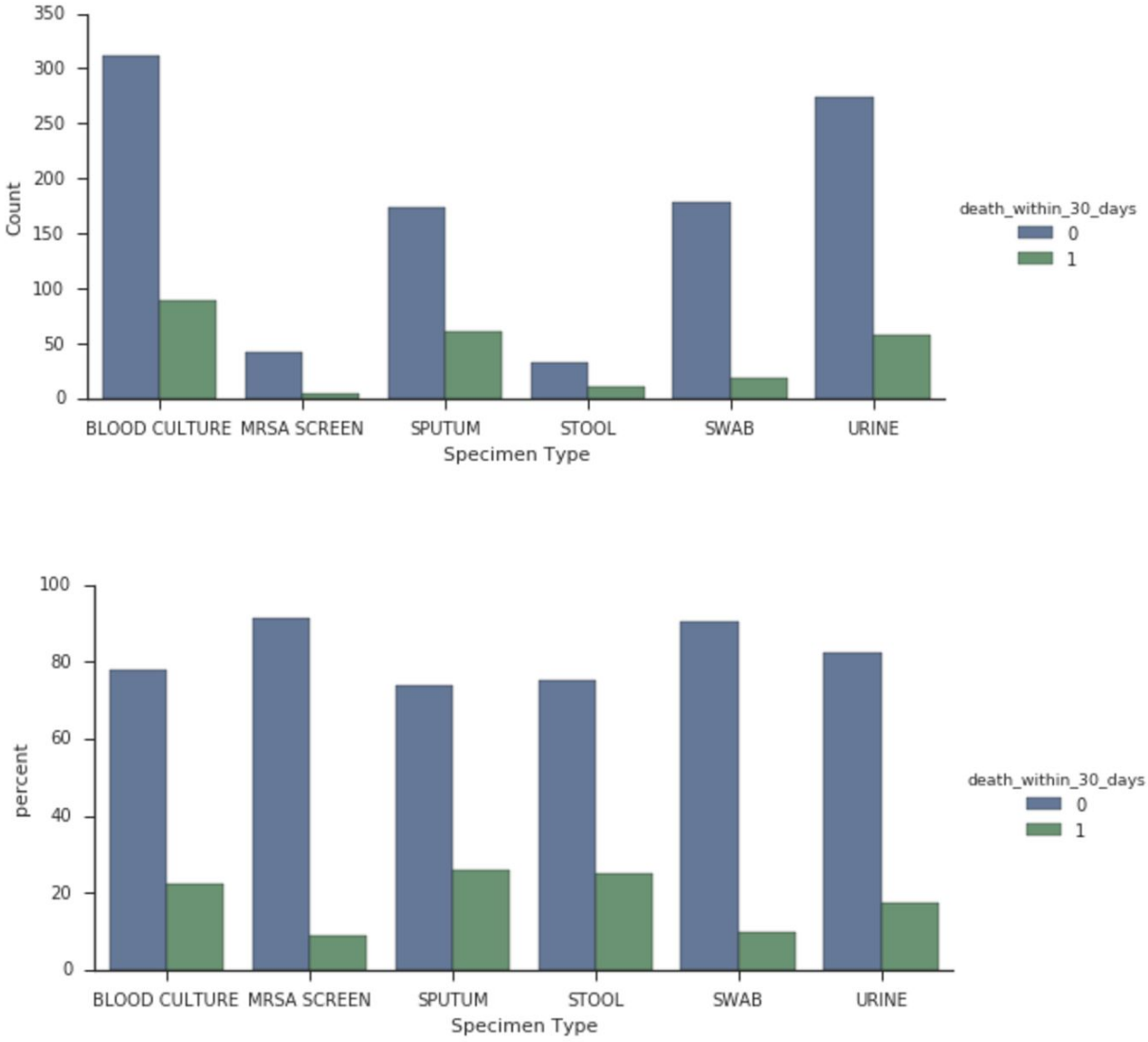
Based upon the analysis done, there is enough data available for option 1 and option 2. But, not enough data available for option 3. So, option 1 and option 2 are used as features for machine learning model.

- **Analyze the data -** Infection data was visualized many different ways to understand impact of this data on survivability -
  - Survivability by number of infections -





- ❏ Survivability by infection Type -



Preparation of Input Data

Demographics, vital (lab results) and infection data are merged together to create a single data set that is fed to machine learning models. During the process, it's found out that there are several null values in the merged data set. Null value for each column is analyzed and an approach is designed to manage them. Here are steps taken for null values for different columns -

1. For the number of infections, null value is converted to zeros. That means if patient has null infection, then patient has no infection.

2. For each specimen type infection, null value was converted to zero. That means if patient has null infection means that patient has no infection.

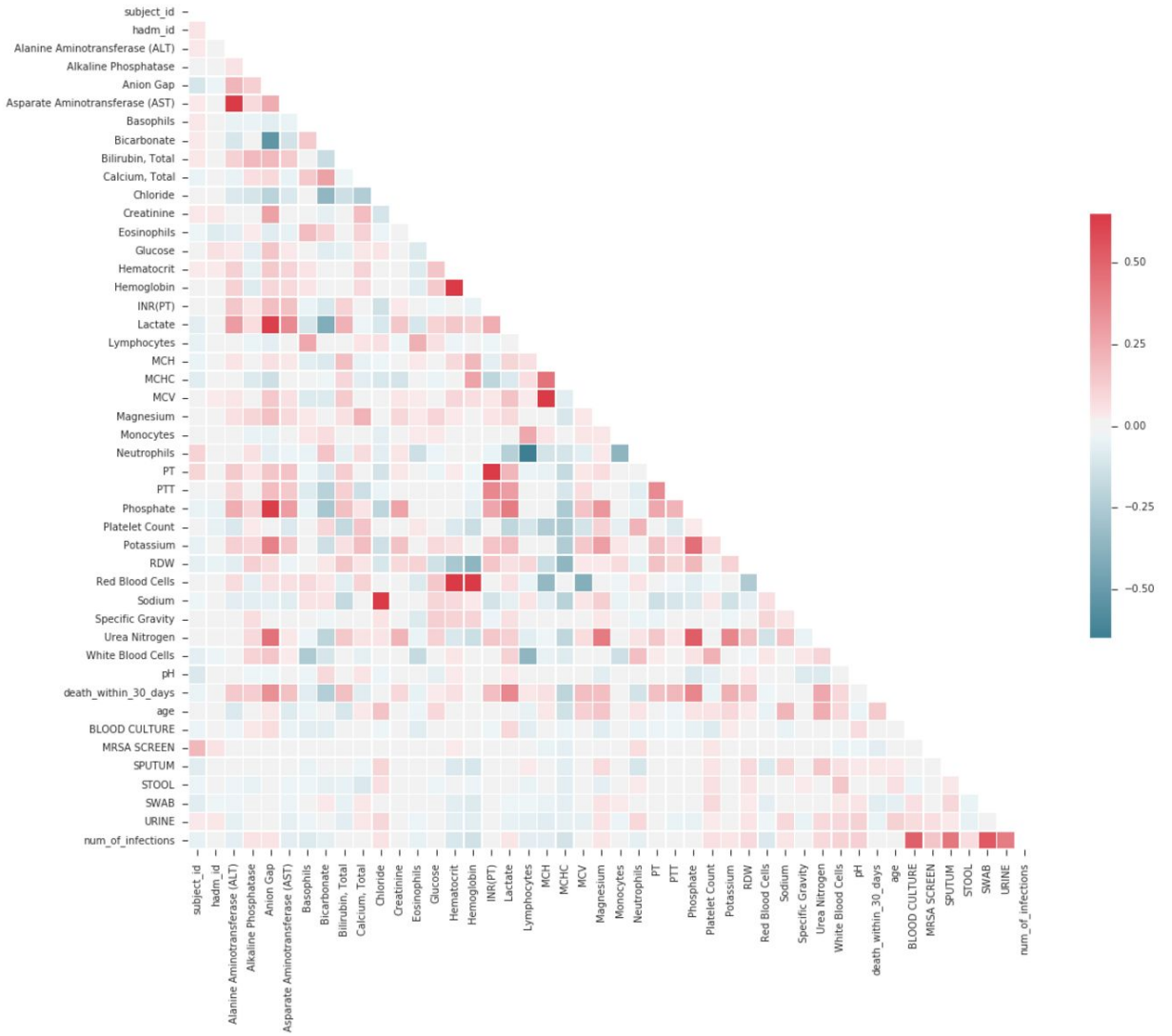
3. For each of the lab result, null value was replaced with mean value of that lab result.

All the data are merged together in a single CSV file ('Final_sepsis_feature_list.csv') and saved such that this data can be used for machine learning models.

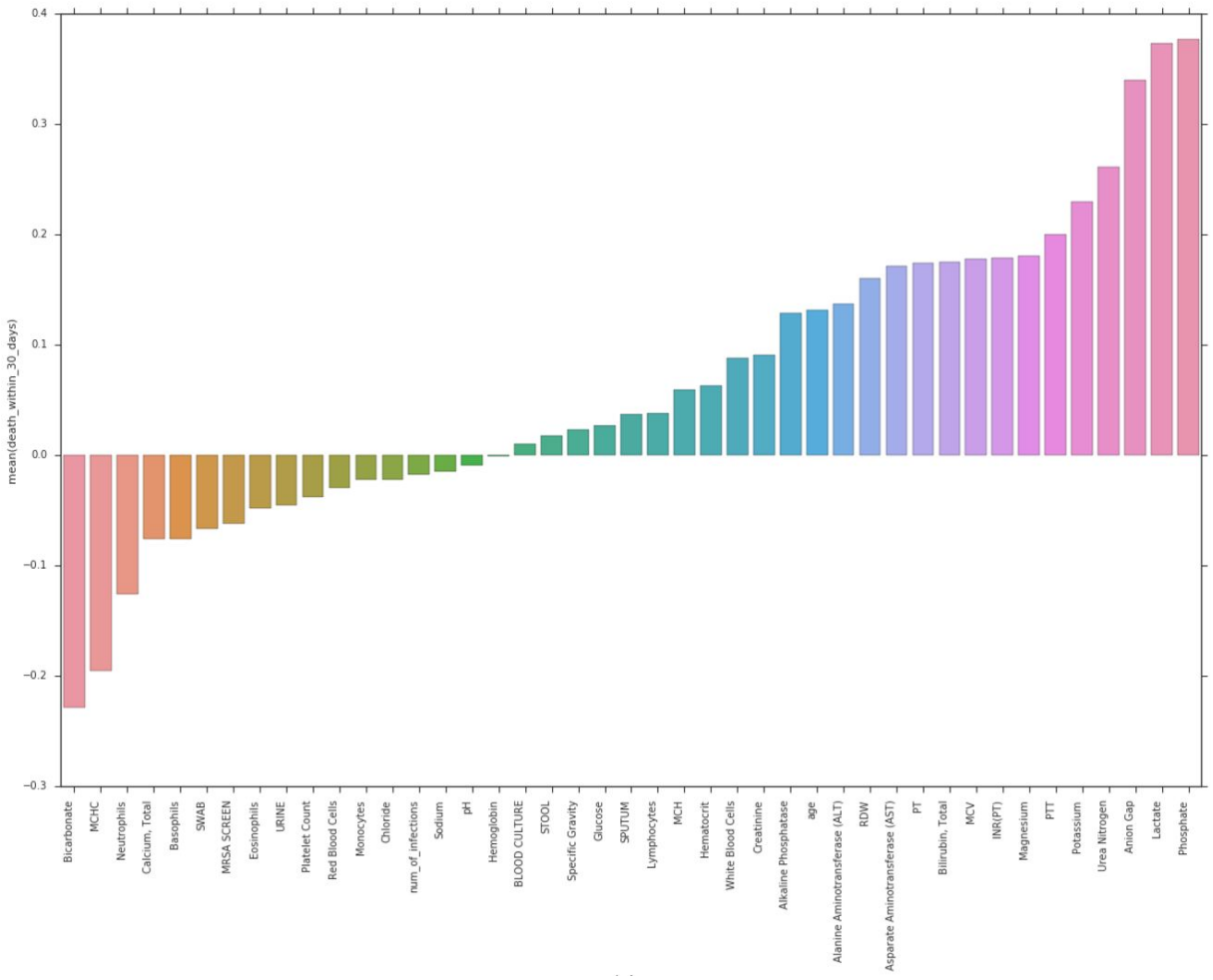
Correlation across data -

Once all data is merged together, correlation among features is analyzed -

- ❏ Feature to feature correlation -



❑ Correlation with survival within 30 days -



Algorithms and Techniques

Goal of this project is to predict if a sepsis shock patient will survive after 30 days of ICU admission. It's a classification problem. In order to solve this classification problem, we will try out multiple machine learning models and figure out which one works the best. We will try out following machine learning algorithms -

1. **Logistic Regression** - Logistic Regression is easy to apply and a solid baseline to compare other more complex methods to. It is widely used in many industries (e.g., called the "standard industry workhorse" in the Mahout docs). Thus, it should be the first thing to start with when looking for an optimal classifier since a more complex model should at least beat LR. Of my three methods, it is the one with high bias and low variance.

2. **Adaboost classifier -** Given that we have a good number of categorical features and we have translated all features to binary, I thought that ensemble based Adaboost will be a good algorithm to pick up. In addition our features didn't have much correlation with each other or with prediction, so thought that algorithim like k-nearest neighbor and Gaussian Naive Bayes will not work very well.

3. **Gradient Boosting Classifier** - GBC allows to go to the limits of what's possible with the given dataset. Since there is not much correlation across our data, GBM has a good chance to find them with the given data. Also since the dataset is small, extensive experiments with hyperparameters are feasible. Given our situation, GBM is the low bias and high variance model I choose.

4. **Support Vector Machine (SVM)** - With complex dataset, SVM is capable of creating more obvious boundaries between the classes. SVM algorithm is also able to compute a much more optimal hyperplane. Given the complex dataset we have, it made complete sense to try out SVM.

5. **Random Forest** - Random forest runtimes are quite fast, and they are able to deal with unbalanced and missing data. With large number of features, Random forest is very good at prioritizing features and picking the right set of features. Also Random Forests is quite versatile and works well in many different scenario. So, it made complete sense to build a model using Random Forest algorithm.

6. **Neural Network** - Neural networks are an incredibly powerful machine learning algorithm. They can approximate any nonlinear function and they are highly customizable. They are pretty immune to outliers as well. With these features, it is possible to create a highly accurate predictor of the data that forms more complex feature relationships.

Benchmark Model

There are 2 types of Benchmark being selected -

1. **Data Driven Naive predictor** - Looking at the distribution of data (those who survive past 30 days, and those who don't survive), it's clear that most patients survive (~80%) past 30 days. This can greatly affect **accuracy**, since we could simply say *"this patient survives past 30 days"* and generally be right, without ever looking at the data! Making such a statement is called **naive**, since we have not considered any information to substantiate the claim. As we build the machine learning model, we will look at Naive

prediction and make sure that our Machine learning model performs better than Naive prediction. Here is the Naive prediction for the patient not surviving past 30 days -

Accuracy: 21.3922%

F-score: 0.2538

2. **Industry research driven** - Quite a few studies have been done to predict survivability of a Sepsis diagnosed patient. I am using the metrics as described in https://www.ncbi.nlm.nih.gov/pubmed/23442987 -

   "Survival after sepsis was predicted with an accuracy of 80% by the NN model, which used only information collected at the time of the diagnosis of sepsis. The development of multiple organ failure after the diagnosis of sepsis was predicted accurately (81.5%) with either the MLR or the NN model. Both the MLR and the NN methods depended on the interpretation of a likelihood quantity, requiring the choice of a threshold to make a survival prediction. The accuracy of the MLR models was very sensitive to the threshold value. The accuracy of the NN models was not sensitive to the choice of threshold, because they generated likelihood predictions that were distributed far from the middle range where the threshold was placed. "

   While this study was done in 1996, there have been few follow-up studies. The few follow-up studies had a similar success rate with slightly different attributes and test periods. Due to the detail of this study, most focus will be put on the work of Flanagan et al. for the benchmark.

Data Preprocessing

- **Normalization of Data -**

Since the vital (or lab results) data has many different ranges, it was required to normalize them before feeding them into machine learning model (such that all features have equal importance). Also since there are a lot of outliers, Robustscaler from sklearn preprocessing module was selected to do the normalization.

- **One hot encoding of categorical data -**

All the categorical data were one hot encoded. Categorical features were admission type, insurance, ethnicity and gender.

- **Shuffle and split data** -

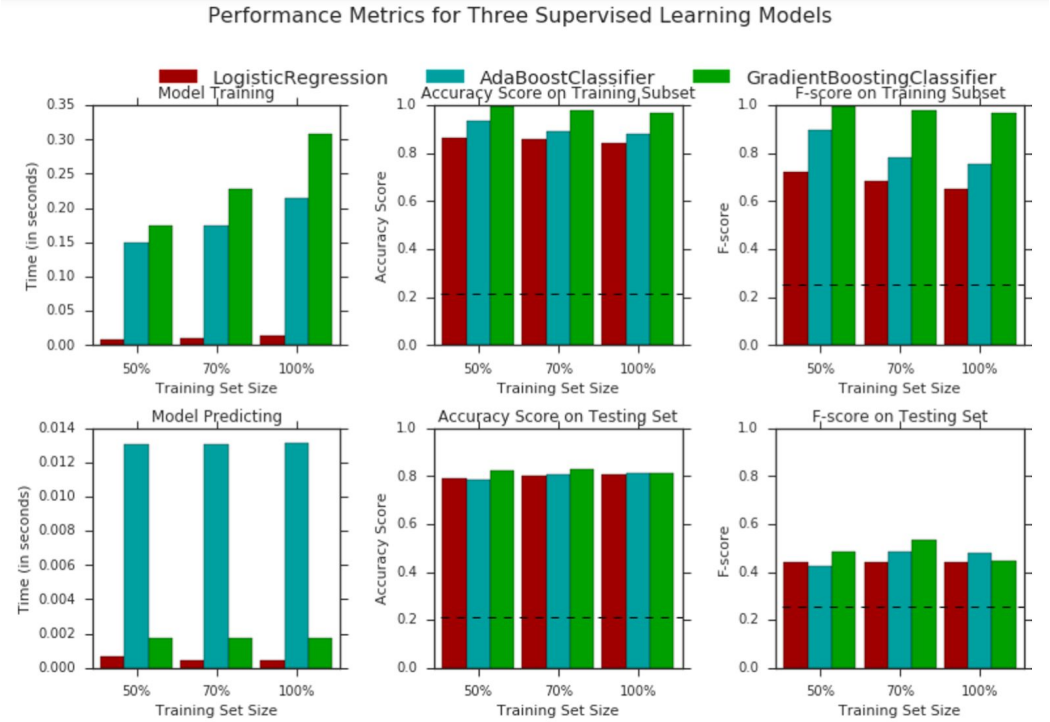
Now that all categorical variables have been converted into numerical features, and all numerical features have been normalized, data is split (both features and their labels) into training and test sets. 90% of the data will be used for training and 10% for testing. We are doing 90% and 10% because our dataset is small.

Implementation of Machine Learning models

1. Using sci-kit learn, following machine learning algorithms were run -
    a. Logistic Regression
    b. Gradient Boosting Classifier
    c. Adaboost Classifier
    d. Gaussian Naive Bayes
    e. Decision Tree Classifier
    f. K-neaerest Neighbor
    g. Stochastic Gradient Descent (SGD) classifier
    h. Random Forest Classifier
    i. SVC

2. Models were trained using 50%, 70% and 100% of training data.

3. Results for each module were visualized -



4. Results from all the machine learning algorithms were compiled together and analyzed -

	model_name	iteration#	acc_test	acc_train	f_test	f_train	pred_time	train_time
2	AdaBoostClassifier	2	0.813559	0.880000	0.480226	0.755814	0.013160	0.214714
20	RandomForestClassifier	2	0.822034	0.993333	0.474453	0.993789	0.002826	0.037807
11	GradientBoostingClassifier	2	0.813559	0.970000	0.448276	0.969388	0.001718	0.308026
26	SVC	2	0.826271	0.860000	0.445545	0.735294	0.030247	0.066734
17	LogisticRegression	2	0.805085	0.843333	0.443787	0.651261	0.000471	0.014476
14	KNeighborsClassifier	2	0.809322	0.853333	0.400000	0.688073	0.063403	0.001988
23	SGDClassifier	2	0.762712	0.800000	0.336788	0.536913	0.000438	0.001637
5	DecisionTreeClassifier	2	0.699153	1.000000	0.268199	1.000000	0.000669	0.025445
8	GaussianNB	2	0.216102	0.300000	0.229885	0.282051	0.001069	0.001473

5. Based upon the analysis of results, following algorithms were selected for further refinement -
    a. Adaboost boosting along with Random Forest Classifier
    b. SVC
    c. Gradient Boosting Classifier

Challenges

For the implementation of the process above, implementing machine learning algorithm itself was easy. But, deciding feature and preparing the data for machine learning model was complex. There were quite a few challenges with the implementation of the project process above -

1. Because there was no domain expert available with this project, it was hard to decide which features to drop or pick. In this case, we at high level figure out what categories of data is needed. Then within those categories, we found out which data are available in sufficient volume. We picked up the features for which sufficient data was available in the dataset. Feature for which enough data was not available was dropped. Before picking and dropping feature, it will be good to get a domain expert feedback whether we are not missing something very important by dropping the features.

2. There were quite a few outliers in the dataset. But since data was limited and sparse, it was not easy to discard the outliers. So for the normalization of data, we picked up

Robustscaler from sklearn preprocessing. This normalization module has better management for outlier data.

3. Dataset was small and features were sparse. 1184 records is just not sufficient enough to come up with a robust machine learning model. In real life scenario, do more investigation to figure out how higher volume of data can be found and dataset is not that sparse. As more and more health care providers now are using EHRs, it should be possible to get a lot more data that is much better populated.

Refinement of Selected Algorithms

Once the machine learning algorithms were short listed, each of the the algorithims was further optimized. Using GridSearchCV module of scikit-learn, all the algorithms were further optimized with different parameters. Here are the parameters that were used for optimization -

a. **AdaBoost boosting with Random Forest classifier**
    i. For Random Forest classifier -
        1. Max_depth : [1,3,5,10,20]
        2. criterion : ["gini", "entropy"]
        3. Max_features : [2,10,15,20,30,40,50]
    ii. For AdaBoost boosting
        1. N_estimators : [1,2,5,10,20,30,80]
        2. Learning_rate : [0.01,0.1,0.5,1.0]

b. SVC classifier
    i. 'C' : [1, 1e1, 1e2]
    ii. 'gamma': [0.0001, 0.0005, 0.001, 0.005, 0.01, 0.1,'auto']
    iii. 'kernel':['linear', 'rbf']

c. Gradient Boosting Classifier
    i. 'Max_depth' : [None,3,5,8,16]
    ii. 'Min_samples_split' : [2,4]
    iii. ‘Max_features’ : [None,5,10,15,20,30]
    iv. 'Min_samples_leaf' : [1,5,11,50]
    v. 'N_estimators':[50,100]
    vi. 'Learning_rate' : [0.01,0.05,0.1]

There were quite a few challenges optimizing the algorithims -

A. It was taking really long time to optimize the algorithm. In order to get superior performance, multithreading using n-jobs was implemented with Gridsearch.

B. Some of the parameters value were taking long time to run, but was not giving optimal performance (specifically with SVM algorithm). For example, C value of 5e3 was taking really long time. Such low performing and time consuming parameters were dropped from the optimization parameters set.

2. A neural network using Tensor flow was also trained. For neural network, following parameters were optimized -
    a. Hidden_num_units
    b. Output_num_units
    c. Epochs
    d. Batch_size
    e. learning_rate

Results

Model Evaluation and Validation

Upon optimization of all the machine learning modules, here is the final result -

Machine Learning Algorithm	Test accuracy	Test F-score
AdaBoost boosting with Random Forest classifier	84.32%	0.5673
SVM	84.75%	0.5714
Gradient Boosting Classifier	83.47%	0.4950
Neural Network	81%	.5022

AdaBoost boosting with Random Forest classifier

Initially, Adaboost boosting and Random Forest Classifier was used separately. During optimization phase, they were combined together. We got major boost in performance after using Random forest to classify and applying Adaboost boosting for weak learner. Also optimization of model gave quite a bit performance boost -

Unoptimized model

Accuracy score on testing data: 0.8263
F-score on testing data: 0.4587

Optimized Model

Final accuracy score on the testing data: 0.8347
Final F-score on the testing data: 0.5369

Best parameters turned out to be -

1. For Random Forest classifier -
    a. Max_depth : 1
    b. criterion : "entropy"
    c. Max_features : 50
2. For AdaBoost boosting
    a. N_estimators : 30
    b. Learning_rate : 0.5

Optimization test was carried out several times with these parameters, we were able to get this result consistently. More training and test data will help further improved the score.

SVM (Support Vector Machine)

With SVC, optimization improved the performance drastically -

Unoptimized model

Accuracy score on testing data: 0.8263
F-score on testing data: 0.4455

Optimized Model

Final accuracy score on the testing data: 0.8474
Final F-score on the testing data: 0.5714

Best parameters turned out to be -

1. 'C' : 100
2. 'gamma': 0.0001
3. 'Kernel': 'rbf'

With the best parameter, test was run several times and same quality of result was obtained consistently. Since feature set was small, no wonder SVM produced the best results.

Gradient Boosting Classifier

Among shortlisted algorithm, this was the worst performer. But that's not surprising with low volume of data. Here is the results with most optimized model -

Unoptimized model

Accuracy score on testing data: 0.8136
F-score on testing data: 0.4483

Optimized Model

Final accuracy score on the testing data: 0.8347
Final F-score on the testing data: 0.4950

1. 'Max_depth' : 16
2. 'Min_samples_split' : 2
3. 'Max_features' : 5
4. 'Min_samples_leaf' : 11
5. 'N_estimators':100
6. 'Learning_rate' : 0.05

Max_Depth is quite high. Need to optimize this model more to achieve better performance.

Neural Network

Neural network gave a decent result, but not great. But, that was understandable. In order to build a good neural network, volume of data needed is quite high. We did put a lot of effort to optimize. Optimization did provide much better performance, but it's still not at acceptable level.

Robustness of models

Goal of this machine networking models is to predict patients who won't survive within 30 days of ICU admission. So, we are going to test how system performs against the patient who die for sure within 30 days of ICU admission. This will give a good picture of how good are the precision of the models we have built - are the models able to give consistent performance for such patients?. Similarly, we will test how model will perform when we know for sure that patient is not going to die within 30 days of ICU admission.

In order to test the above, we broke the test dataset into 2 categories -

- patients who died within 30 days of ICU admission
- patients who survived for more than 30 days

Here are the results of tests -

Machine Learning Algorithm	Test for patients who didn't survive past 30 days of admission	Test for patients who did survive past 30 days of admission
AdaBoost boosting with Random Forest classifier	35.56%	95.81%
SVM	26.67%	98.43%
Gradient Boosting Classifier	22.22%	97.91%
Neural Network	55.56%	85.34%

Here are a few observations from the test results above -

1. Note that all the models do really well when patients are likely to survive past 30 days of admission. But, models don't do well when patients are likely to die within 30 days of admission. Still, all models do much better than Naive predictor of 19% accuracy.
2. Although neural network scores low on overall accuracy and F-score, it does much better job of predicting Test for patients who die within 30 days of admission.
3. Considering we had only 207 patients data for patients who didn't survive, this test result was not surprising. We will need a lot more patient data to train the model to achieve superior accuracy.

All in all, I won't say that model is robust yet. It doesn't provide accuracy consistently across all kinds of data yet. We will need to train the model with a lot more data to make it more robust.

Justification

All the models outperformed the benchmark model. Although results are promising, it could have been much better if we didn't have following limitations -

1. The dataset consists of data for 1184 patients. To obtain a more reliable results, dataset with several hundred thousand patients will be necessary.

2. Data for several infections and lab tests were either missing or were invalid.

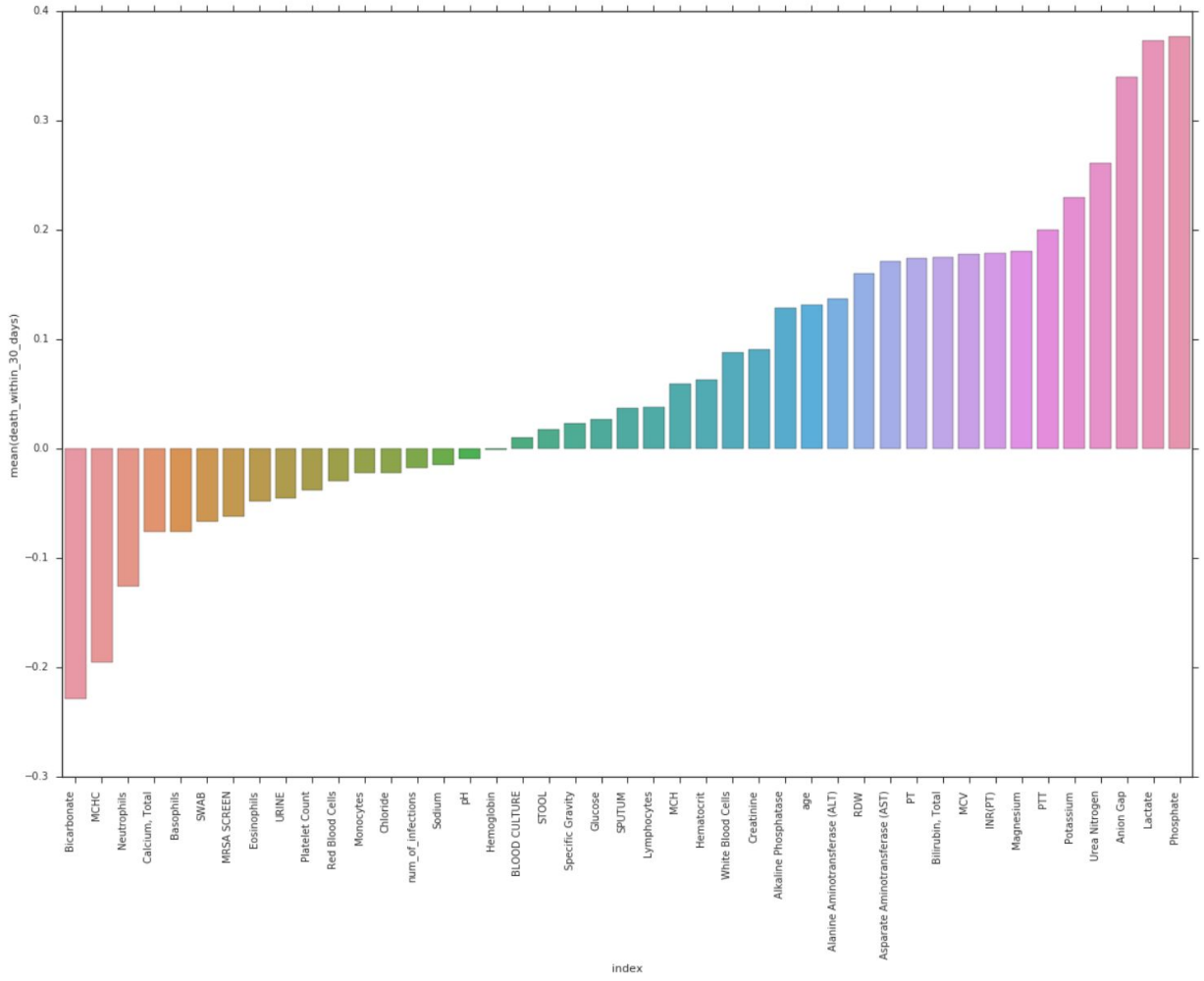
3. There was no input from somebody with domain expertise on Sepsis infections. We may not have missed some important features outside vitals and infection that's must have to predict survivability due to sepsis shock.

Conclusion

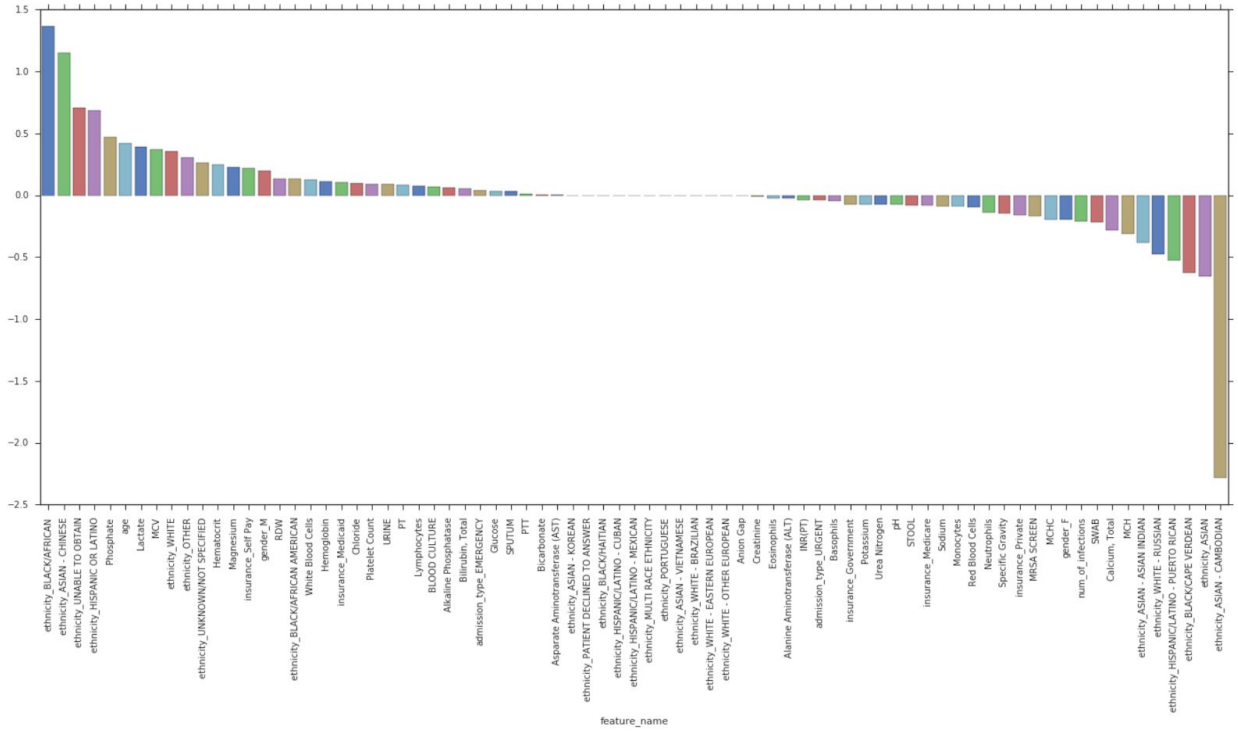
Free-Form Visualization

For free-form visualization, I will like to present how the feature importance (or correlation) of target vary between actual data and across machine learning models. Here are different scenarios -

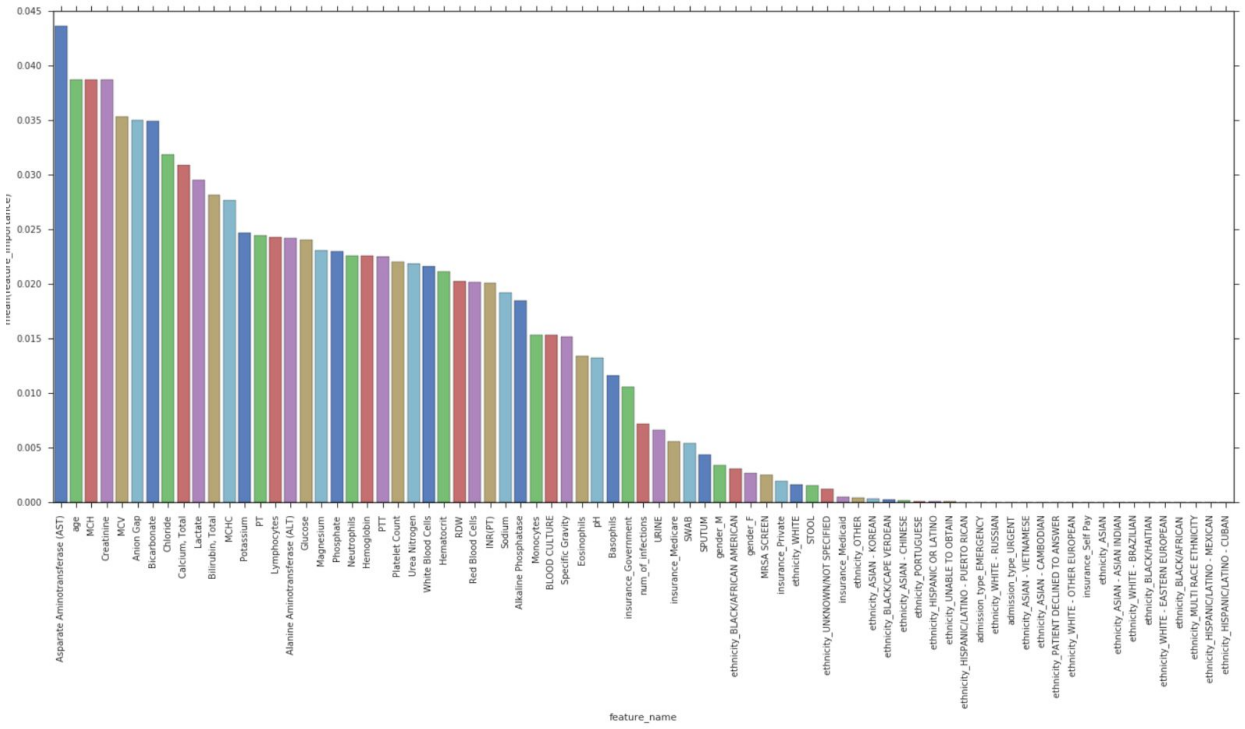
- **Actual correlation between features and the target**:



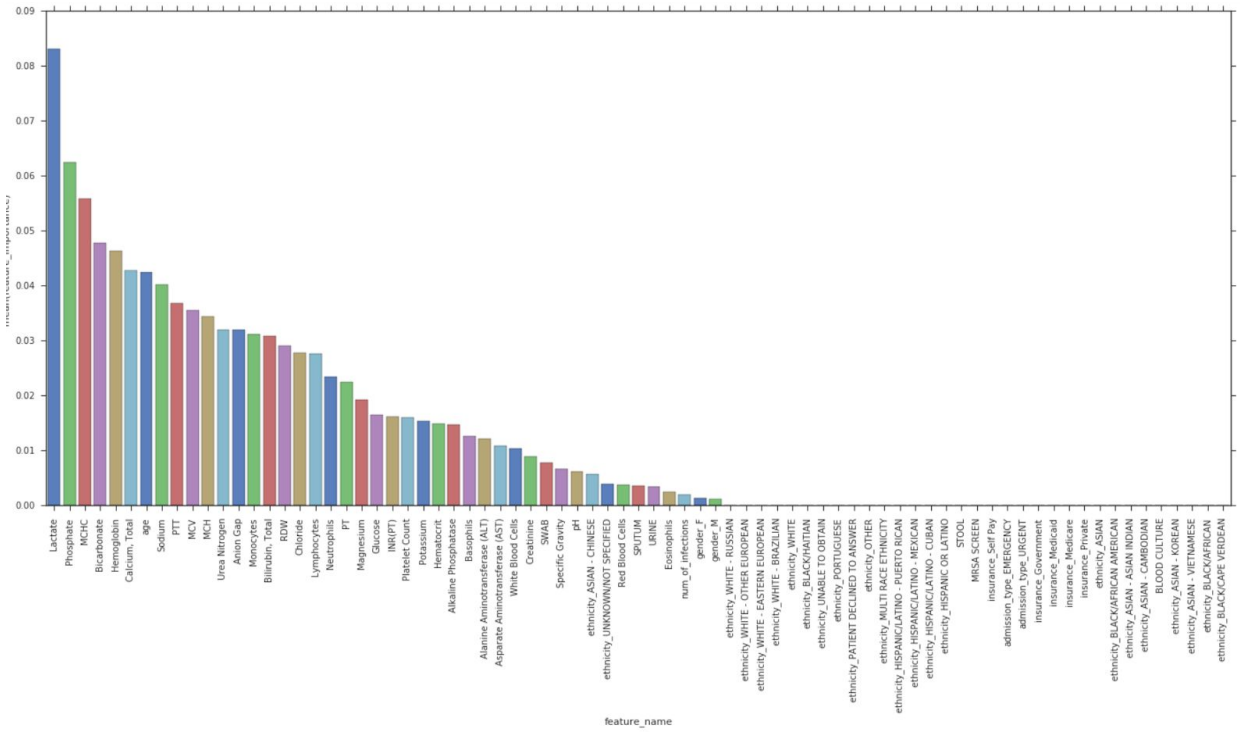
- Feature importance based upon SVM machine learning algorithim



- **Feature importance based upon AdaBoost boosting with Random Forest classifier**



- **Feature importance based upon Gradient Boost classifier**



- **Top 10 features by each method (importance high to low)-**

Actual Correlation	SVM feature importance	Adaboost with Random Forest Feature importance	Gradient Boost Feature importance
Phosphate	Ethinicity	Asparate Aminotrasferase (AST)	Lactate
Lactate	Phosphate	Age	Phosphate
Anion Gap	Age	MCH	MCHC
Urea Nitrogen	Lactate	Creatine	Bicarbonate
Potassium	MCV	MCV	Hemoglobin
Bicarbonate	MCH	Anion Gap	Calcium
PTT	Calcium	Bicarbonate	Age
MCHC	Hematocrit	Chloride	Sodium
Magnesium	Magnesium	Calcium	PTT

It's interesting to note how feature importance matches and vary across different models. Here are some of the key observations -

1. Feature importance and correlation are closest among Gradient Boosting classifier and actual correlation. It's furthest on Adaboost with Random Forest classifier.
2. All machine learning models treat age as quite an important feature.
3. Only SVM regards Ethnicity as the feature and that too most important feature.

So, all in all feature importance varies quite a bit among different machine learning models as well as in actual correlation based upon data. That's why it's important not to drop the features by just viewing the co-relation with target. In order to decide feature priorities, it's important to run different machine learning models and learn from them.

Reflection

The process followed to execute on this project was following -

1. Select a general topic where machine learning can be applied to study (in this case health care space was selected).

2. Figure out the dataset that's needed to build machine learning model. In this scenario - based upon the selected problem (survivability of Sepsis patients) , MIMIC-III database was selected.

3. A deeper research was done to understand the problem and the kind of data needed to predict survivability of Sepsis patients. Then selected dataset was analyzed to understand how required data can be retrieved from the selected dataset.

4. Once dataset was understood - target feature was figured out. In this case, we had to calculate the target field (i.e. patients who died within 30 days of admission).

5. All features were analyzed and prepared such that they could be used as input to machine learning models.In this case, following categories of features are analyzed to figure out the right set of features to build machine learning model -
    a. Patient's age, gender, insurance information etc.
    b. Patients vitals collected within 48 hours of admission
    c. Patients infection that's reported within 24 hours of admission

6. Once features are finalized and all the data are available, multiple models using different Machine learning algorithm are implemented, analyzed and a comparative analysis between them are done. Based upon the comparative analysis, results are published.

There were quite a few challenges with execution of the project above -

4. Because there was no domain expert available with this project, it was hard to decide which features to drop or pick. In this case, we at high level figure out what categories of data is needed. Then within those categories, we found out which data are available in sufficient volume. We picked up the features for which sufficient data was available in the dataset. Feature for which enough data was not available was dropped. Before picking and dropping feature, it will be good to get a domain expert feedback whether we are not missing something very important by dropping the features.

5. Dataset was small and features were sparse. 1184 records is just not sufficient enough to come up with a robust machine learning model. In real life scenario, do more investigation to figure out how higher volume of data can be found and dataset is not that sparse. As more and more health care providers now are using EHRs, it should be possible to get a lot more data that is much better populated.

All in all, process and methodology used in doing the project above could be used in several health care scenario. With better quality of data (i.e. data that's less sparse) and strong input from health care domain experts, several very practical prediction models can be built using above processes and methodology.

Improvement

For me, this project is not over yet - it's just a beginning. I will make this project better by doing following -

1. First of all, I will like to get a lot more Sepsis related patient data than it was available with MIMIC-III database. 1184 patients data is just not sufficient to come up with a very accurate model. Now that EHR systems have been adopted by several health care providers, it will be possible to get a lot more data than just 1184 patient records.

2. I will like to find out dataset that's not as sparse as Mimic-iii dataset. With higher adoption of EHR and EMR systems, health care providers now have data tracked a lot more consistently than early 2000s data of MIMIC-III database.

3. I will work closely with Doctors to understand importance of different features. Then based upon that understanding, I will tweak some of the features to get better results.

4. Now that we understand feature importance, I will run machine learning model with fewer features and also apply L1 and L2 regularization.

5. With dense dataset, I will also come up with a better management of outliers data. Better management of outliers may have a strong impact on performance of the machine learning models.

Reference

1. Wikipedia - https://en.wikipedia.org/wiki/Sepsis
2. Epidemiology of severe sepsis - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3916382/
3. Machine learning may help in early identification of severe sepsis - https://www.sciencedaily.com/releases/2017/05/170524100616.htm
4. Sepsis and Machine Learning - https://vision.cloudera.com/sepsis-and-machine-learning/
5. Predicting In-Hospital Mortality due to Sepsis: An Integrative Approach - https://repository.library.brown.edu/studio/item/bdr:697404/PDF/
6. Cloud-based Analytics: Supporting Healthcare's Digital Transformation - https://d1.awsstatic.com/whitepapers/Industries/HCLS/AWS_WhitePaper_21216.pdf
7. Predicting survival of patients with sepsis by use of regression and neural network models - https://www.ncbi.nlm.nih.gov/pubmed/10156949