Title

SepTrack: An efficient sepsis-onset tracker built using machine learning techniques to monitor high-risk patients

Introduction

Sepsis is a life threatening condition that occurs as a result of the body's response to infection causing tissue damage and even death. Internationally, an estimated 30million people develop sepsis and 6 million people die from sepsis each year; an estimated 4.2million newborns and children are affected. Hospital acquired sepsis is even worse as patients can contract the disease even after the primary ailment is cured. In India the mortality rate of ICU acquired sepsis is very high [1]. Large scale epidemiological studies [2] have shown that clinical parameters can be attributed to high mortality rate among ICU patients.

The inherent variability of the disease further complicated the analysis. A patient might arrive at the hospital with a septic shock condition or might be diagnosed during the stay. The diagnosis procedure to confirm or deny the presence of sepsis is very complex and often relies on the clinician's judgement. A machine learning framework to predict risk of mortality for sepsis patients will be very helpful. This will help clinicians take early decisions and treat the patient at the very onset of the disease. If sepsis is not treated early, multiple organ dysfunction might occur and with each hour the survival rate also falls drastically.

Objectives

A predictive model will be very helpful in detecting high-risk patients. After a patient is admitted and the blood tests and clinical data (like Insurance, Admission type and Gender) are collected, we will build predictive models to classify survivability of patients within 30 days of admission. This is the most critical period for surviving sepsis/septic shock infection.

We considered MIMIC III (Medical Information Mart for Intensive Care III) free hospital database. This database contains de-identified data from over 50,000 patients who were admitted to Beth Israel Deaconess Medical Center in Boston, Massachusetts from 2001 to 2012. In order to get access to the data for this project, you will need to request access at this link (https://mimic.physionet.org/gettingstarted/access/).

1. Use the clinical features and lab measurements that are available within 24 hours of admission as features.
   The following were the features used:
   - Age, Admission type, Insurance, Gender
   - Bicarbonate levels, INR, MCH, AST, creatinine levels, platelet count, glucose, hemoglobin, lactate levels etc
2. Use the discharge clinical notes to extract valuable features and build predictive models
3. Combine clinical+lab+notes based features to improve prediction accuracy

Dataset

The analysis used four datasets from the MIMIC III database. This requires separate access and the data can't be public. But since this data is needed by the contest reviewers to judge my model's performance, I am creating a Google Drive link to download to share the data privately. Kindly refrain from sharing the link with anyone. The names of the four files required to carry on the analysis are the same as mentioned in the Google drive folder. ***This is the link.***

Analysis

- **Data exploration**: We used the MIMIC III (Medical Information Mart for Intensive Care III) free hospital database. This database contains de-identified data from over 50,000 patients who were admitted to Beth Israel Deaconess Medical Center in Boston, Massachusetts from 2001 to 2012. In order to get access to the data for this project, you will need to request access at this link https://mimic.physionet.org/gettingstarted/access/).
  We downloaded the ADMISSIONS table, PATIENTS table, LABEVENTS table and the NOTEEVENTS table for our purposes. A snapshot and description of each of the tables is shown below.

  - ADMISSIONS table: this contains the patient admission details like sample id, hospital admission id, admission type, admission location, insurance type, etc.

	ROW_ID	SUBJECT_ID	HADM_ID	ADMITTIME	DISCHTIME	DEATHTIME	ADMISSION_TYPE	ADMISSION_LOCATION	DISCHARGE_LOCATION	INSURANCE
0	21	22	165315	2196-04-09 12:26:00	2196-04-10 15:54:00	NaN	EMERGENCY	EMERGENCY ROOM ADMIT	DISC-TRAN CANCER/CHLDRN H	Private
1	22	23	152223	2153-09-03 07:15:00	2153-09-08 19:10:00	NaN	ELECTIVE	PHYS REFERRAL/NORMAL DELI	HOME HEALTH CARE	Medicare
2	23	23	124321	2157-10-18 19:34:00	2157-10-25 14:00:00	NaN	EMERGENCY	TRANSFER FROM HOSP/EXTRAM	HOME HEALTH CARE	Medicare

  - PATIENTS table: contain patient details like DOB, hospital stay id etc

	ROW_ID	SUBJECT_ID	GENDER	DOB	DOD	DOD_HOSP	DOD_SSN	EXPIRE_FLAG
0	234	249	F	2075-03-13	NaN	NaN	NaN	0
1	235	250	F	2164-12-27	2188-11-22 00:00:00	2188-11-22 00:00:00	NaN	1
2	236	251	M	2090-03-15	NaN	NaN	NaN	0

  - LABEVENTS table: Lab measurements associated with the patients along with the date and time.

	ROW_ID	SUBJECT_ID	HADM_ID	ITEMID	CHARTTIME	VALUE	VALUENUM	VALUEUOM	FLAG
0	281	3	NaN	50820	2101-10-12 16:07:00	7.39	7.39	units	NaN
1	282	3	NaN	50800	2101-10-12 18:17:00	ART	NaN	NaN	NaN
2	283	3	NaN	50802	2101-10-12 18:17:00	-1	-1.00	mEq/L	NaN

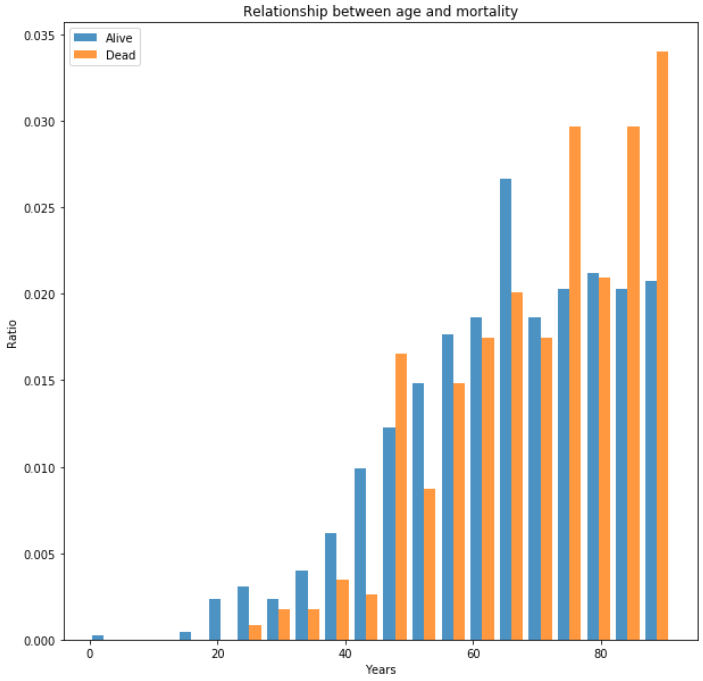
- NOTEEVENTS table: This table contains the free text clinical notes entered by the physician.
- **Exploratory data analysis**

    - We considered only discharge notes of patients for further analysis. Mortality period was calculated by subtracting deathtime from admittime. The labels for the dataset were ***0 (those patients who will survive after 30 days) or 1 (those patients who will die within 30 days).*** The risk of a patient dying within 30 days of admission is the dependent variable that we are trying to predict.

	ROW_ID	SUBJECT_ID	HADM_ID	CHARTDATE	CHARTTIME	STORETIME	CATEGORY	DESCRIPTION	CGID	ISERROR	TEXT
0	174	22532	167853.0	2151-08-04	NaN	NaN	Discharge summary	Report	NaN	NaN	Admission Date: [**2151-7-16**] Dischar...
1	175	13702	107527.0	2118-06-14	NaN	NaN	Discharge summary	Report	NaN	NaN	Admission Date: [**2118-6-2**] Discharg...
2	176	13702	167118.0	2119-05-25	NaN	NaN	Discharge summary	Report	NaN	NaN	Admission Date: [**2119-5-4**] D...

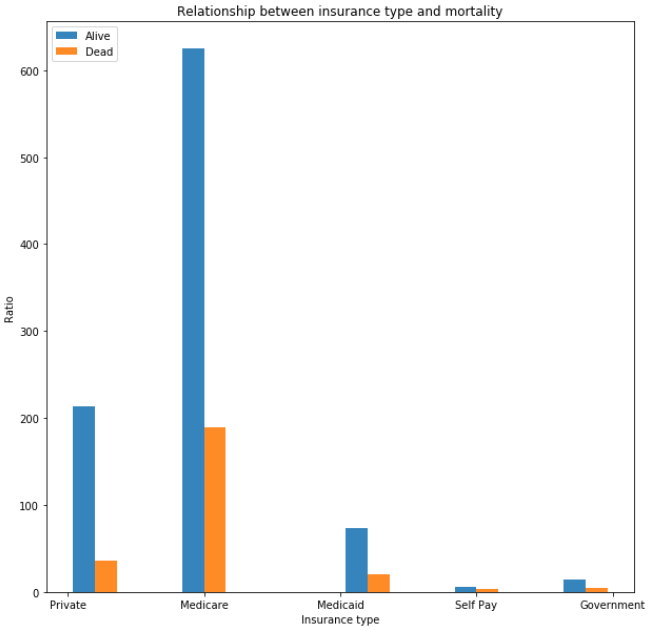
    - **Age** was not explicitly mentioned as a feature in any of the tables. So, we constructed it ourselves by subtracting ADMITTIME from DOB. Then we did some basic EDA to understand the structure of our data and get useful insights about the features.

    - **Relation between age and mortality rate.**



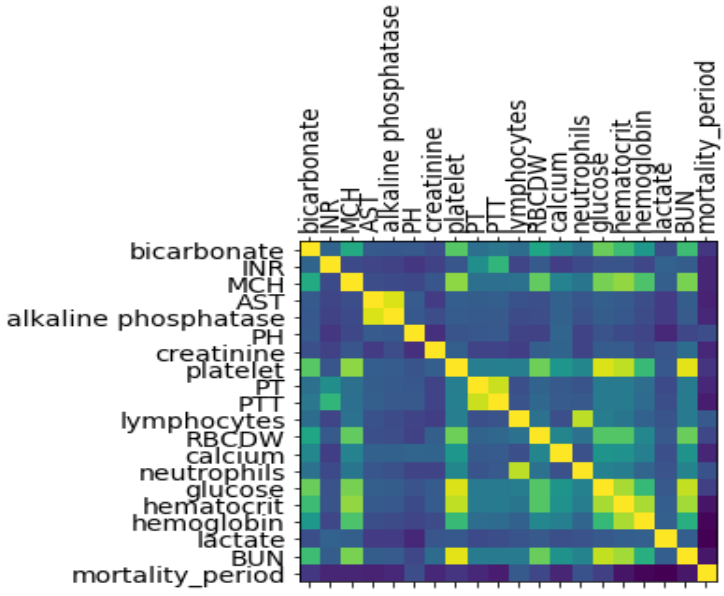
From the graph it's very evident that older patients have a greater chance of dying of sepsis/septic shock within 30 days than their younger counterparts.

- **Relation between insurance type and mortality**



In the above graph, Medicare users constitute a large chunk of people who either survive or die. Most Medicare users are funded by the government and constitutes a large section of older people. The younger patients tend to be more open to private insurance schemes than older people. Hence the disparity. Medicaid is for people who are below the poverty line. Although absent from the database, the income feature should play a huge role in survivability.

The lab events table was subsetted by the admission id and only those measurements that are taken within 24 hours of admission are included.
A correlation analysis was done between the lab measurements and mortality rate and the results are as follows:



From the correlation plot, we can clearly see that mortality period is highly correlated with lactate levels [3], Blood urea nitrogen levels [4], hemoglobin, creatinine [5], AST [1] , MCH [2] etc. This has solid literature support and some of them (like serum lactate) is used as clinical biomarkers to detect sepsis.
Imputation: Some missing values for the lab tests were replaced with mean of that respective column.

- **The noteevents table**
  This is in part the most interesting part of the analysis. The noteevents table contains free text clinical notes from discharge summaries. Basic preprocessing (like replacing na values with empty string ‘ ’ etc) was applied on each note. There were almost 0.02% of notes missing that were filled with blank spaces.
  We can’t directly input the strings as features and they need to be converted to a form that can be used for model building. All the sentences were tokenzed into individual words and punctuations and numbers were replaced with spaces. The tokens were all converted to lower case letters for ease of analysis.

  CountVectorizer: The CountVectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. It works as follows:
  1. Create an instance of the CountVectorizer class.
  2. Call the fit() function in order to learn a vocabulary from one or more documents.
  3. Call the transform() function on one or more documents as needed to encode each as a vector.

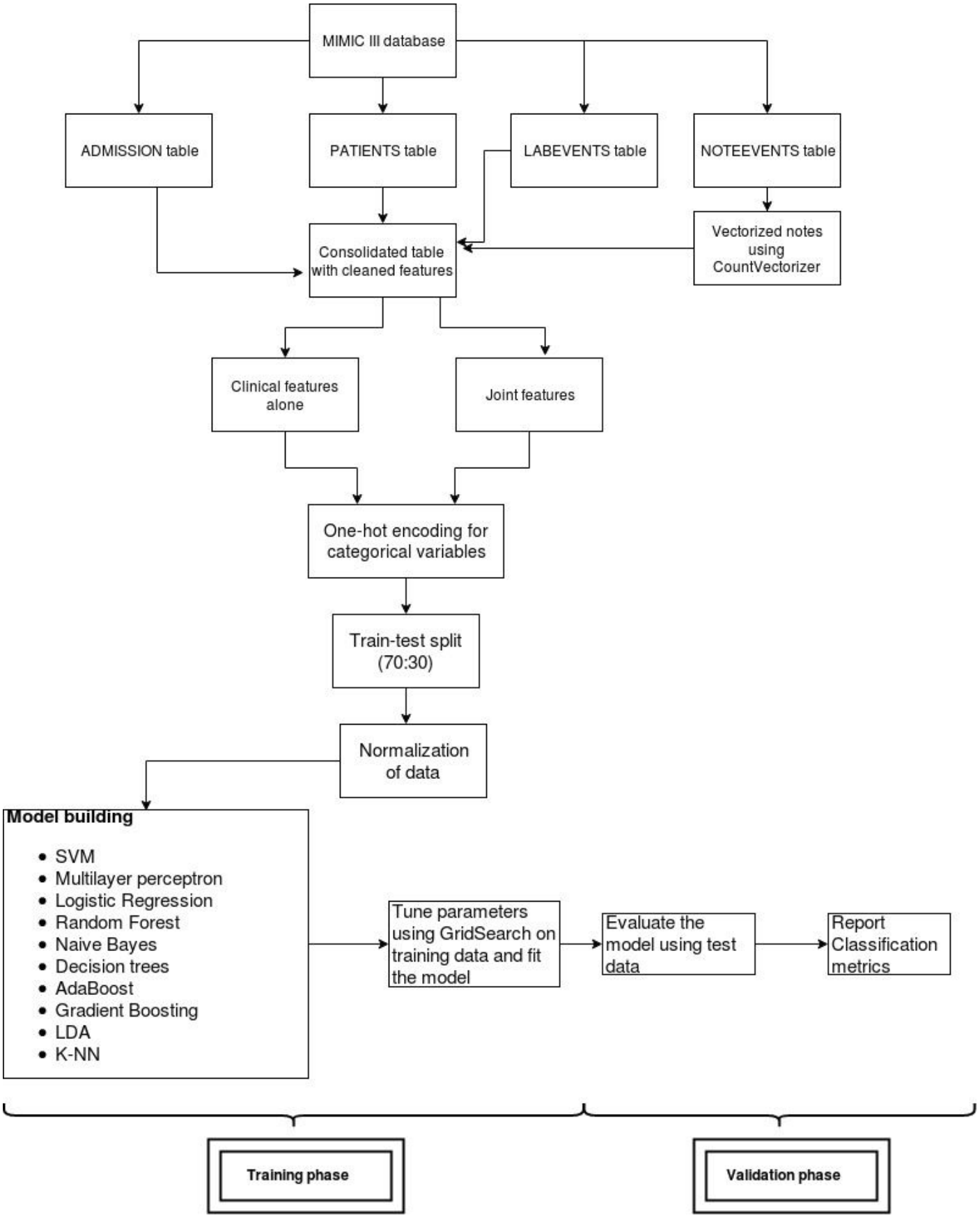
  As an example consider you have three sentences in your corpus:
  “I love dogs”, “I hate dogs and knitting”, “Knitting is my hobby and my passion”
  Then the vectorized way to represent them would be:

	I	love	dogs	hate	and	knitting	is	my	hobby	passion
Doc 1	1	1	1							
Doc 2	1		1	1	1	1				
Doc 3					1	1	1	2	1	1

(Image source: Link)

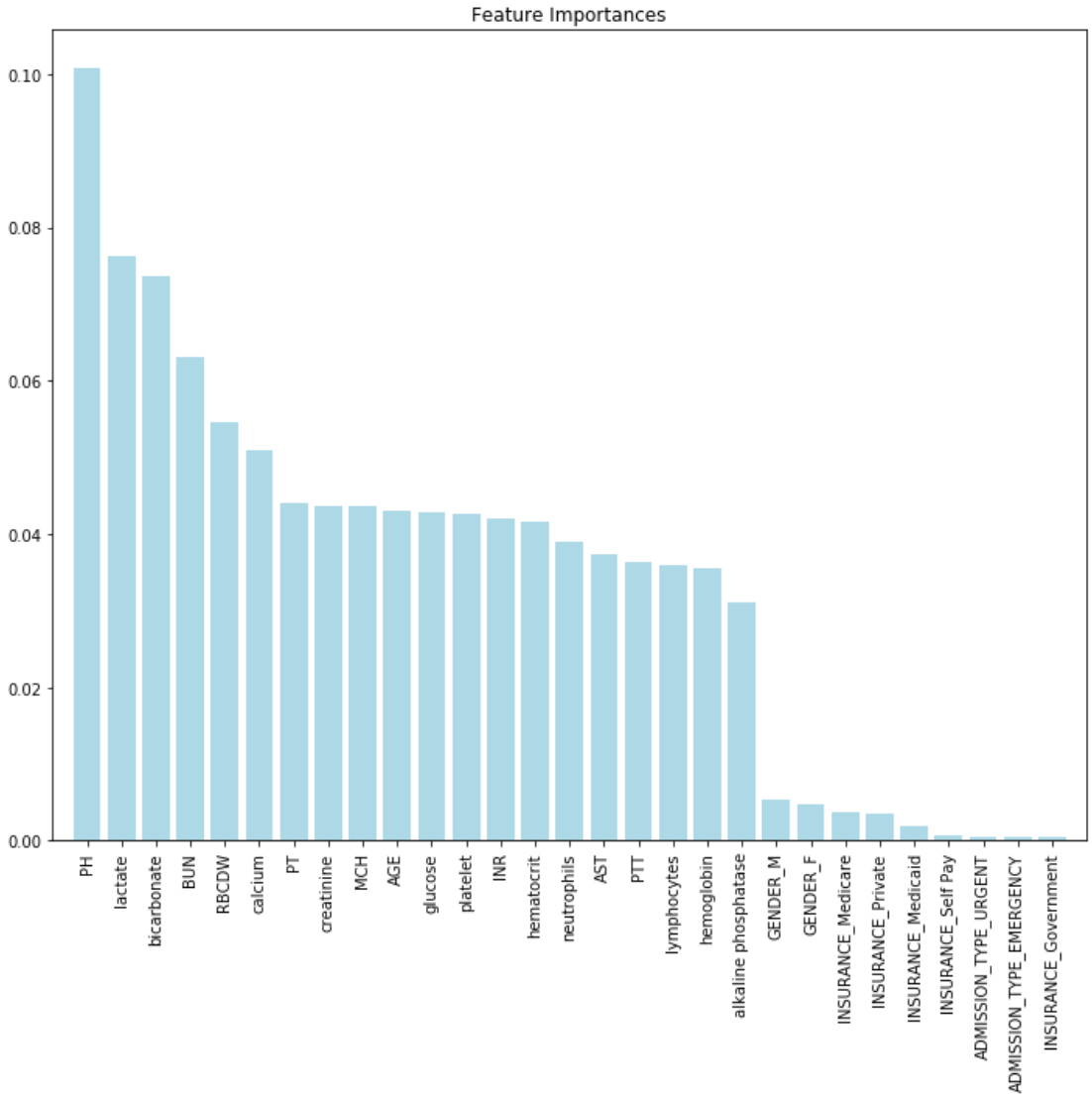
The entire notes feature was vectorized using the CountVectorizer and the corresponding vectors for each note was saved for further use. So in total, we have four cleaned dataframes (admission, patients, labevents and notes). They were all combined into one by considering hospital admission id (HADM_ID) as the joining column

Flowchart for analysis



Model Building

Feature selection: The importance of the clinical features for further analysis is shown below:



From the above plot, it is very clear that Gender, Insurance and admission type are not really predictive enough to be included on our further analysis. PH, lactate levels etc are highly important for predicting survivability and will be included in the analysis.

Results: Clinical features alone

10 fold cross validated AUC

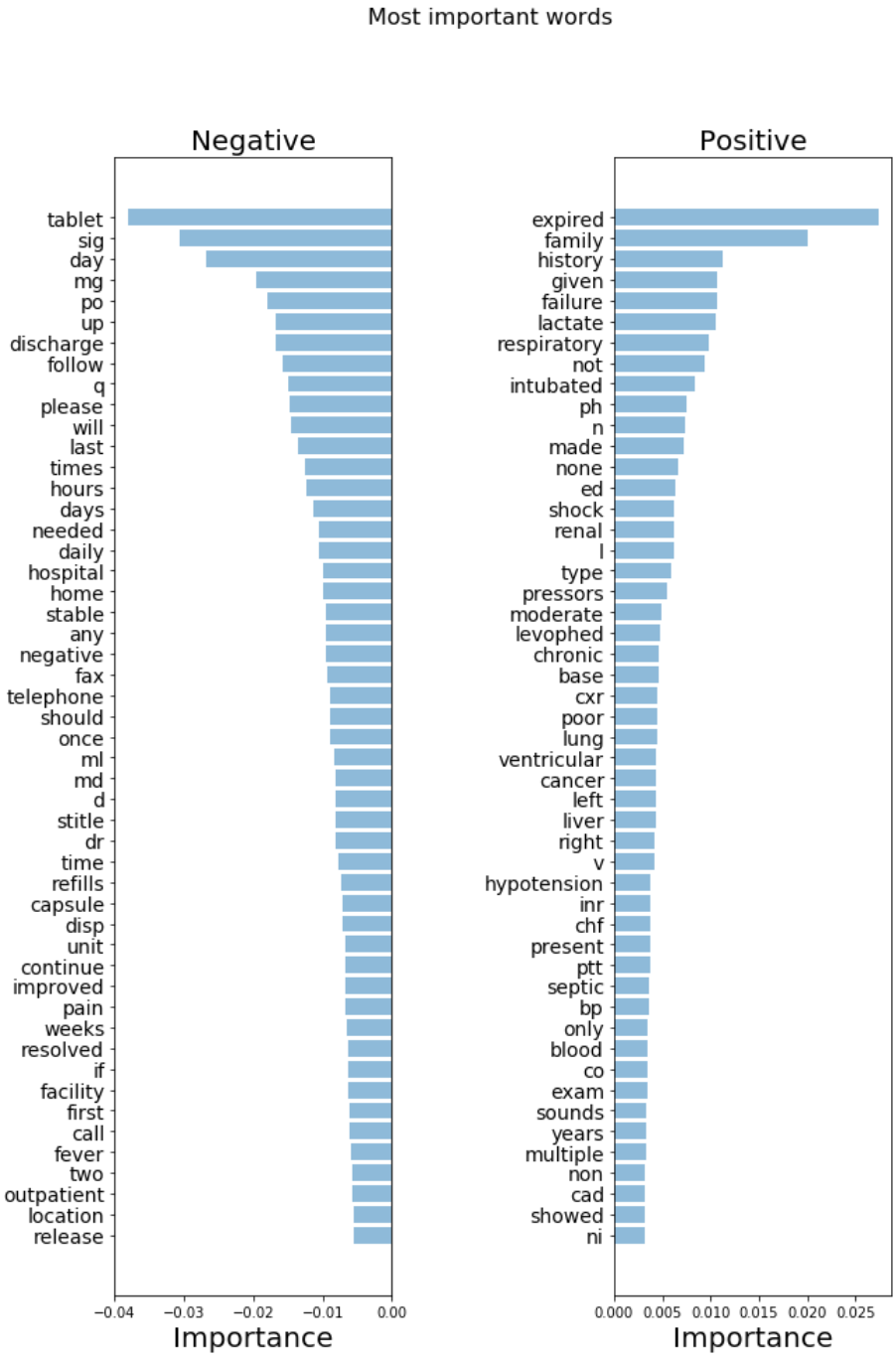
Classifier	Best parameters	AUC
SVM	C=100, gamma =0.001, kernel= rbf	0.7837
Random forests	Criteria =gini, max depth =7 number_of_estimators=600	0.8163
Decision Trees	max_depth=5, min_samples_split =110	0.750
Logistic Regression	C=0.464, penalty='l1'	0.788
Naive Bayes	----	0.657
AdaBoost	base_estimator_criteria=gini	0.757
Gradient Boosting	criterion: 'friedman_mse', learning_rate: 0.15, max_depth: 5 max_features: 'sqrt', min_samples_split: 0.245, n_estimators: 10	0.779

Results: Text features alone

10 fold cross validated AUC

Classifier	Best parameters	AUC
SVM	C=1, gamma=0.001, kernel=rbf	0.978
Random forests	Criteria =entropy, max depth =7 number_of_estimators=400	0.988
Decision Trees	max_depth=7, min_samples_split=190	0.968
Logistic Regression	C=0.1, penalty=l1	0.984
Naive Bayes	---	0.902
AdaBoost	base_estimator_criteria=gini	0.984
Gradient Boosting	criterion= 'friedman_mse', learning_rate =0.15	0.982

Results: Important words



The most important string features that were essential in predicting <30 day survivability is shown above. Here the positive class represents patients who died within 30 days and vice versa. As expected, the words describing the positive class like “expired”,”history”,”respiratory”,”failure”,”lactate” [3] ,”intubated”,”PH” (due to high PH levels in sepsis patients), levophed (which is favored as the first-line vasopressor for **septic** shock),

CHF (congestive heart failure) etc are very relevant to sepsis mortality. This further proves the textual model's efficacy in capturing essential words for sepsis patients. A similar pattern can be observed for the negative class (patients who didn't die within 30 days) as well.

Data Analytics Acceleration Library

For purposes of my work, I used the Intel's DAAL framework to speed up computation. Since, I have implemented the cross validation results without the DAAL framework, I split the dataset into train/test and then implemented it using the DAAL framework and reported the test results. I created an intelpython environment using conda and installed daal4py inside that. Next, I launched the jupyter notebook from inside the environment to get its full functionality.

Since this is an imbalanced problem (#minority class=252 and #majority class=923), I applied SMOTE as a means to oversample the minority class. The exact order of operations so as to prevent data leakage were as follows:

1. First split the data into train/test
2. Then apply SMOTE only on train and keep the test seperate
3. The fit models on the oversampled train data
4. Predict the labels of the test data using the fitted model above
5. Report the classification metrics

Results: Using clinical features alone

Support Vector Machines - Test Results

Class	Precision	recall	F1 score	AUROC
Survived	0.97	0.88	0.93	0.8655
Deceased	0.53	0.85	0.65	

The reason for denoting class-wise metrics is that, this is a class imbalance problem and using overall accuracy and other such metrics is misleading. So class wise breakdown shows a decent recall for the minority class (Deceased, i.e 0.85) which is the target of interest in most healthcare analytics problem. This shows that the model is capturing 0.85 fraction of the true deceased individuals based on clinical data alone.

Results: Using text features alone

Support Vector Machines - Test Results

Class	Precision	recall	F1 score	AUROC
Survived	0.98	0.92	0.95	0.90
Deceased	0.69	0.89	0.78	

The reason for denoting class-wise metrics is that, this is a class imbalance problem and using overall accuracy and other such metrics is misleading. So class wise breakdown shows a decent recall for the minority class (Deceased, i.e 0.90) which is the target of interest in most healthcare analytics problem. This shows that the model is capturing 0.90 fraction of the true deceased individuals based on clinical data alone.

Results: Using both clinical and text as features

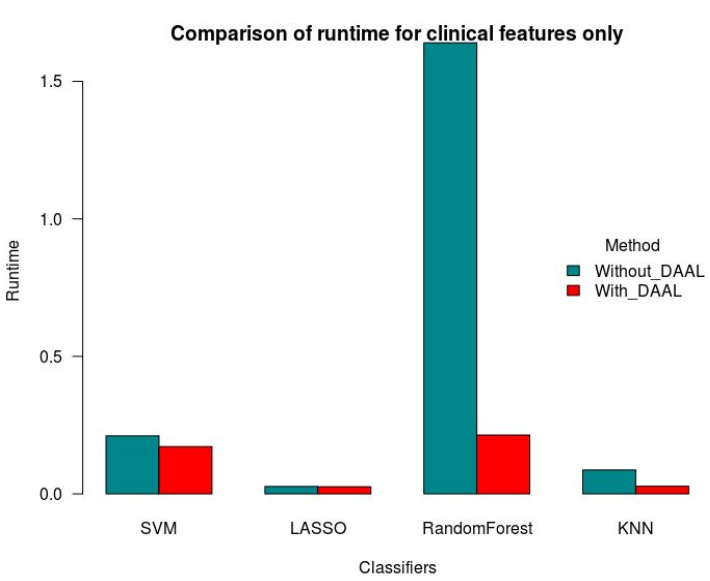
Support Vector Machines - Test Results

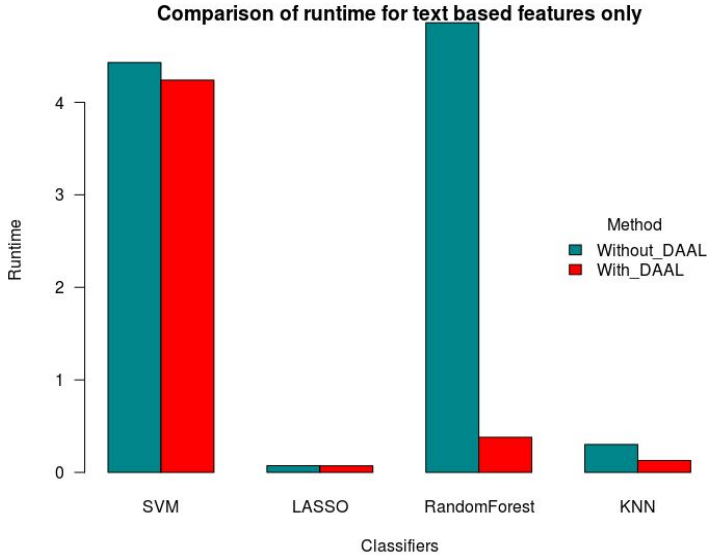
Class	Precision	recall	F1 score	AUROC
Survived	0.98	0.87	0.92	0.878
Deceased	0.46	0.88	0.60	

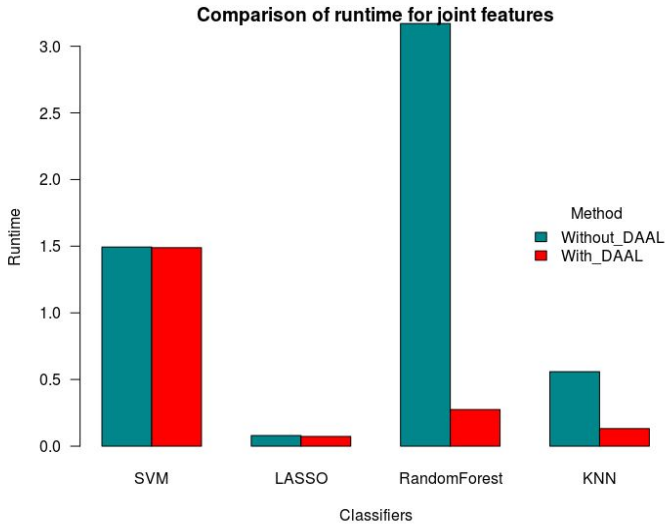
The classifier built using the text as features proved to be a better model than the one built using clinical data alone and the one built using both clinical and text features.

The results of using four different classifiers (SVM, LASSO, Random forest, K-nearest neighbors) and the comparison of their runtimes with or without using DAAL is present in the notebook daal4py_implementation.ipynb.

Comparison of runtime for different classifiers







The above three plots show the runtimes for different classifiers using sklearn (without DAAL) and with_DAAL respectively. It is very evident that the trend is similar across all three feature sets. The most noticeable improvement is observed for Random forest and KNN while the least is for LASSO and SVM. In any case, the DAAL library gives a significant boost to the total runtime as compared to sklearn.

Benchmarks

The following benchmark was considered for my purposes of validation/performance measure:

> *From vital signs to clinical outcomes for patients with sepsis: a machine learning basis for a clinical decision support system" by Eren Gultepe, et al. [6]*

In the above study, a 0.73 AUC for mortality prediction in patients with sepsis was achieved with only three features: median of lactate levels, mean arterial pressure, and median absolute deviation of the respiratory rate.
The best AUC obtained in my case was 0.984 which is a 34% increase over the initial benchmark. But the primary reason behind this is the number of features that I am using to predict. I am extracting around 500 word features whereas they have used only three. So this should not be misconstrued as *"performing better than the benchmark".* But I wanted to try out text based classification for sepsis mortality and the results are given above.

Discussion

This project was an attempt to utilize free text clinical notes and lab measurements to predict within 30 day mortality of patients.
There is an important observation to be made here. The best AUC for clinical features was 0.81 and that of text features was 0.98. The text features used were the discharge notes and I chose this method to judge the notes' predictive power. But, a person who has died will have words like "dead", "deceased" "expired" written on the discharge notes. So the model's job to discriminate the two classes becomes very easy. Hence the high AUC. My purpose was to see how much discriminatory power the model possess. But for all practical purposes, we should look more closely towards that clinical measurements which are available within 24 hrs of admission. **Hence the AUC of 0.81 (10 fold cross validation without sampling) and 0.8655 (test AUROC using oversampling) should be the overall result of my model that I would like to report. That is a 10.9% and 18% improvement over the previous benchmark respectively [6].**

Conclusion

According to the National Institute of General Medical Sciences, over 1 million people in the United States develop severe sepsis each year, and 15–30 percent of these people die as a result. Other studies estimate that sepsis may contribute to over 250,000 deaths every year.
In India, 34% of sepsis patients die while still in the ICU. Several worldwide efforts [7] are now undergoing in order to understand the disease and come up with a risk score to identify patients early. The above study is one such example to identify the best set of predictive features and stratify high risk patients. The models built are cohort specific and hence data from the Indian cohort is needed to make it relevant to the Indian healthcare sector.
I sincerely thank the organizing team of the Intel Python Hackfury 2019 for their attempt to encourage us to build machine learning models that are relevant to various real-life scenarios.

References

1. Wang D, Yin Y, Yao Y. Advances in sepsis-associated liver dysfunction. *Burns Trauma*. 2014;2(3):97–105. Published 2014 Jul 28. doi:10.4103/2321-3868.132689
2. Muady GF, Bitterman H, Laor A, Vardi M, Urin V, Ghanem-Zoubi N. Hemoglobin levels and blood transfusion in patients with sepsis in Internal Medicine Departments. *BMC Infect Dis*. 2016;16(1):569. Published 2016 Oct 13. doi:10.1186/s12879-016-1882-7
3. Lee SM, An WS. New clinical criteria for septic shock: serum lactate level as new emerging vital sign. *J Thorac Dis*. 2016;8(7):1388–1390. doi:10.21037/jtd.2016.05.55
4. Nusshag C, Weigand MA, Zeier M, Morath C, Brenner T. Issues of Acute Kidney Injury Staging and Management in Sepsis and Critical Illness: A Narrative Review. *Int J Mol Sci*. 2017;18(7):1387. Published 2017 Jun 28. doi:10.3390/ijms18071387
5. Doi K, Yuen PS, Eisner C, et al. Reduced production of creatinine limits its use as marker of kidney injury in sepsis. *J Am Soc Nephrol*. 2009;20(6):1217–1221. doi:10.1681/ASN.2008060617
6. Gultepe E, Green JP, Nguyen H, Adams J, Albertson T, Tagkopoulos I. From vital signs to clinical outcomes for patients with sepsis: a machine learning basis for a clinical decision support system. J Am Med Inform Assoc. 2014;21(2):315–325. doi:10.1136/amiajnl-2013-001815
7. Shimabukuro DW, Barton CW, Feldman MD, et al Effect of a machine learning-based severe sepsis prediction algorithm on patient survival and hospital length of stay: a randomised clinical trial BMJ Open Respiratory Research 2017;4:e000234. doi: 10.1136/bmjresp-2017-000234

Team details
Name: Shayantan Banerjee
Team name: Charaka
Affiliation: IIT Madras
