Project Report

Early Prediction of Sepsis from Clinical Data

Authors: Omer Nahum, Omer Madmon
GitHub: https://github.com/omer6nahum/SepsisPrediction

Introduction

Sepsis is a life-threatening medical condition caused by an immune response to a serious infection. Every year about 7.1 million people in the United States develop sepsis and about 2,000,000 of them die. Early prediction of sepsis and starting medical treatment accordingly are critical and can save many lives. In this assignment we are asked to suggest and evaluate multiple approaches and algorithms for solving the early prediction task, given a time series data of 20k patients recorded while in ICU, containing medical and demographic features.

We will start by exploratory data analysis to better understand the dataset used for the prediction task, including understanding the features distributions, correlations and missing data, including hypothesis testing regarding differences between patients who developed sepsis and healthy patients.

Then, we decide on imputation policy for different features based on the EDA, and create statistical features based on the raw features in order to summarize the time dimension for each patient while still preserving as much information as possible. This way we represent each patient in a fixed dimension vector space.

We then decide on four algorithms to optimize and evaluate for the prediction task: logistic regression, kernel SVM, random forest and XG-boost. We have trained and optimized hyper parameters on the given train set and evaluated on the given test set. Our main result was obtained by XG-boost, having ~72% F1 on the test set. We also used feature importance techniques to better understand decisions made by learned models, which will be visualized in this report as well.

Exploratory Data Analysis

Attached next as a Jupyter notebook (view in GitHub):

Exploratory Data Analysis

Sepsis Prediction

Data Analysis and Visualization Lab - Spring 2022

Omer Nahum , Omer Madmon

Load Data

A single patient i is represented by a $T_i * F$ matrix, where T_i is the number of timesteps recorded for the i'th patient, and F is the number of features (fixed across all patients).

	HR	O2Sat	Temp	SBP	MAP	DBP	Resp	EtCO2	BaseExcess	HCO3	...	WBC	Fibrinogen	Platelets
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN
1	61.0	99.0	36.44	124.0	65.0	43.0	17.5	NaN	NaN	NaN	...	NaN	NaN	NaN
2	64.0	98.0	NaN	125.0	64.0	41.0	27.0	NaN	NaN	NaN	...	NaN	NaN	NaN
3	56.0	100.0	NaN	123.0	65.0	41.0	9.0	NaN	NaN	NaN	...	NaN	NaN	NaN
4	66.0	99.0	NaN	120.0	67.0	43.0	23.0	NaN	NaN	NaN	...	NaN	NaN	NaN
5	94.0	100.0	36.22	194.0	116.0	66.0	14.0	NaN	NaN	NaN	...	NaN	NaN	NaN
6	58.0	99.0	NaN	133.0	68.0	43.0	13.0	NaN	NaN	NaN	...	NaN	NaN	NaN
7	57.0	100.0	NaN	118.0	62.0	37.0	18.0	NaN	NaN	NaN	...	NaN	NaN	NaN
8	62.0	100.0	NaN	126.0	66.0	37.0	12.0	NaN	NaN	NaN	...	NaN	NaN	NaN
9	58.0	95.0	36.11	143.0	77.0	47.0	11.0	NaN	NaN	22.0	...	11.0	NaN	158.0

10 rows × 41 columns

The last column ("SepsisLabel") is a target column, whereas all others are features.
We devide the 40 features into 3 main feature-groups:

- Vital Signs
- Lab Values
- Demographic

We will further address these groups separately, as they tend to share many properties.

Load all patients' psv files:

```
100%|██████████████████████████████████| 20000/20000 [02:05<00:00, 159.33it/s]
```

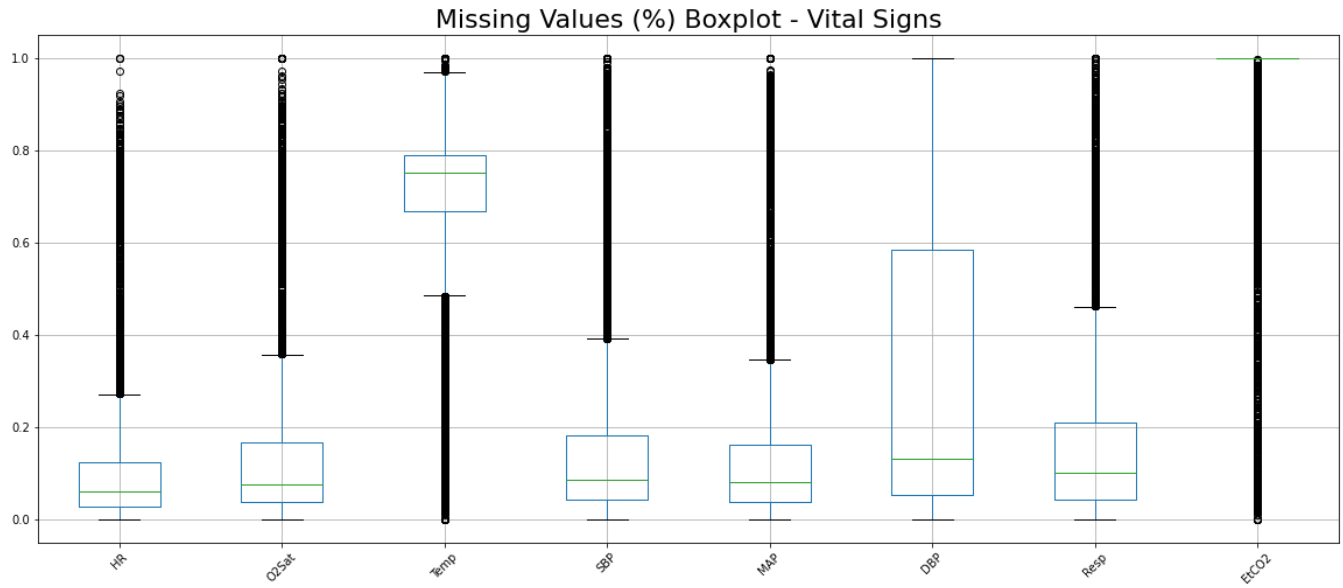
```
Number of patients: 20000
Number of records: 766884
Mean Number of records per patient: 38.3442
```

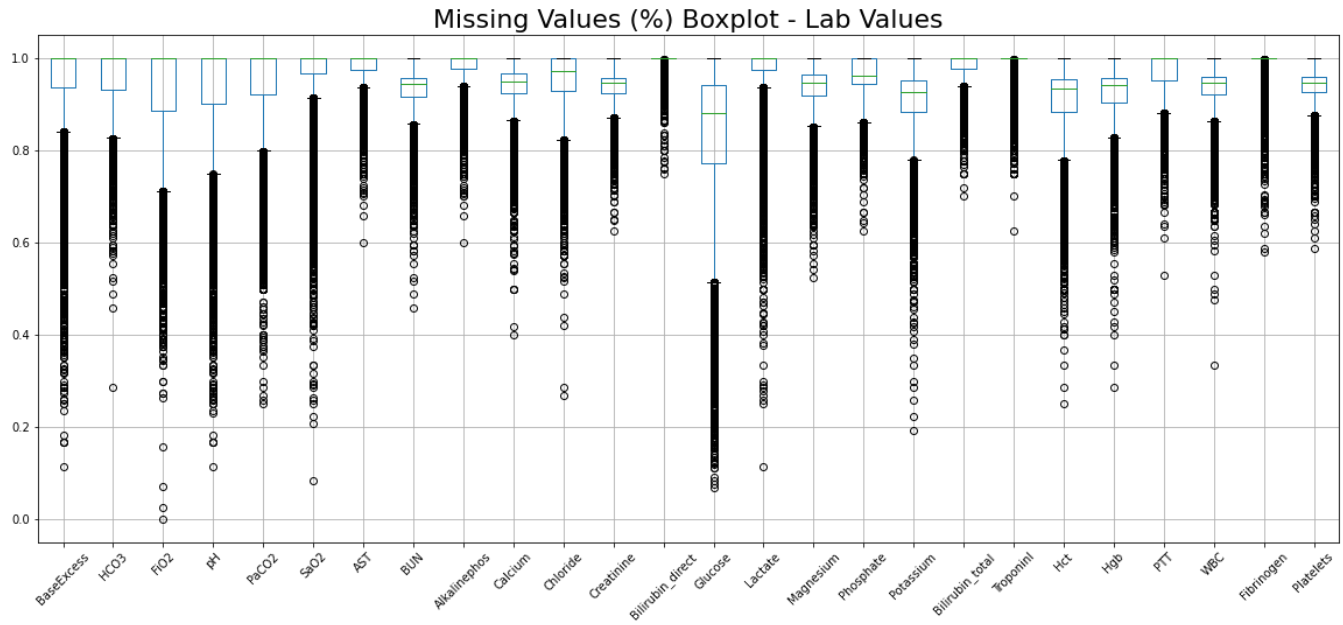
Missing Data

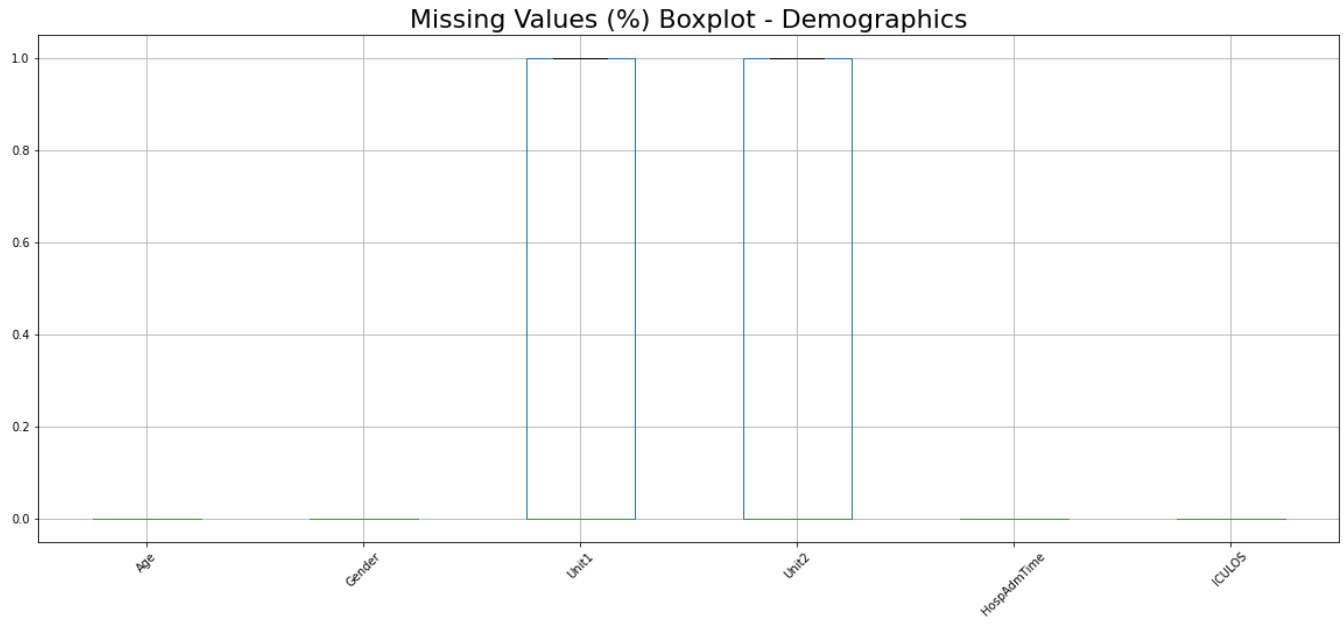
In the following cells we intend to check the precentage of NaNs overall in the data, and specifically per paetient,
in order to better understand the missingness properties of our features, and decide on imputation and features selection methods.

```
100%|██████████████████████████████████| 20000/20000 [03:15<00:00, 102.55it/s]
```

In the following plots, for each group of features (as described above) we visualize boxplots of missing values distributions across patients. In other words, for each feature j we created a vector (of number-of-patients length), where the i'th entry in this vector will be the precentage of missing values of patient i in feature j, and then we plot the distribution for each feature.







From these plots we can clearly observe different missing value propery for each group:

- For most vital signs features, most patients have low precentage of missing value (this can be seen by the location of the box). Vital signs are frequently measured among patients, and therefore as expected have low precentage of missing data.
- For most lab values, the picture is opposite, where most patients have high precentage of missing values. There are 26 different lab values recorded in this dataset (and many more in reality), and these are taken for specific patietns in specific timesteps. Therefore, a high precentage of missing values is expected. Although we barely observe these features for most patients, the exsistence of such features can indicate that a doctor found this lab test important in this timestep.
- For the demographic features, except Unit1 and Unit2, we see that we have (almost) no missing values.

Descriptive Statistics

Analyse distribution of features.

```
Precentage of Sepsis-diagnosed: 7.075%
```

First observation: data is highly imbalanced. Therefore, our evaluation metrics will have to take this into consideration. We will also prefer to weight our classes when using a classification model.

```
5000it [03:10, 26.20it/s]
```

We created a single dataframe, containing all timesteps from a sample of patients (25% of all patients). We will analyse features in this flattened form, instead of per-patient analysis.

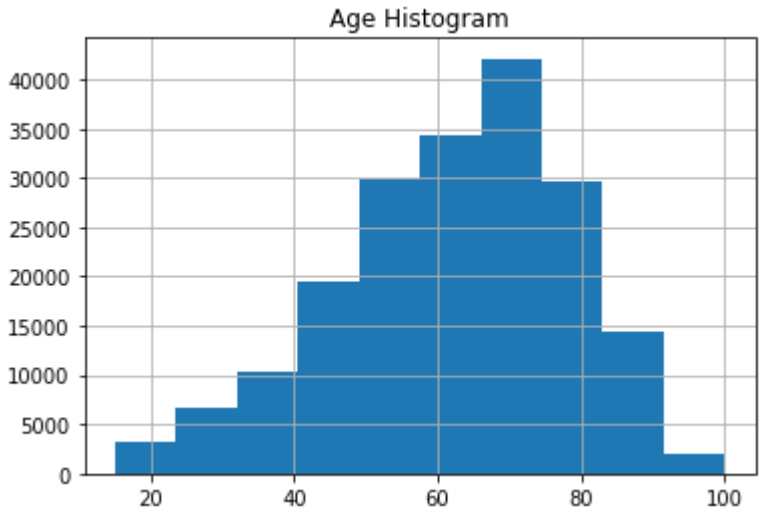
Features description:

	HR	O2Sat	Temp	SBP	MAP	DBP	Resp
count	173327.000000	167638.00000	65892.000000	164440.000000	168352.000000	133198.000000	162716.00000(
mean	84.443762	97.15370	36.986315	123.777075	82.373567	63.814032	18.72214
std	17.371737	2.94011	0.766245	23.480223	16.437753	14.035999	5.08574
min	22.000000	20.00000	20.900000	30.000000	20.000000	20.000000	1.00000(
25%	72.000000	96.00000	36.500000	106.500000	71.000000	54.000000	15.00000(
50%	83.000000	98.00000	37.000000	121.000000	80.000000	62.000000	18.00000(
75%	95.000000	99.00000	37.500000	139.000000	92.000000	72.000000	21.50000(
max	186.000000	100.00000	50.000000	296.000000	298.000000	296.000000	100.00000(

8 rows × 40 columns

For the demographic features of Age and Gender:

Age distribution:



Gender distribution:

```
Precentage of male: 55.647%
```

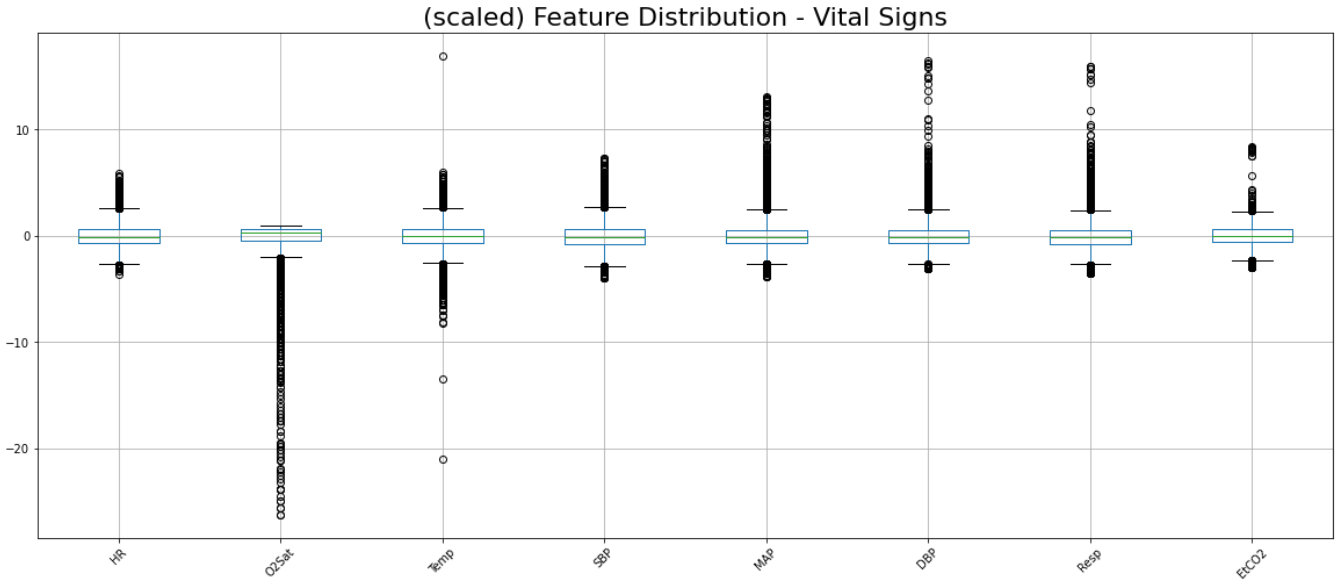
Scaling

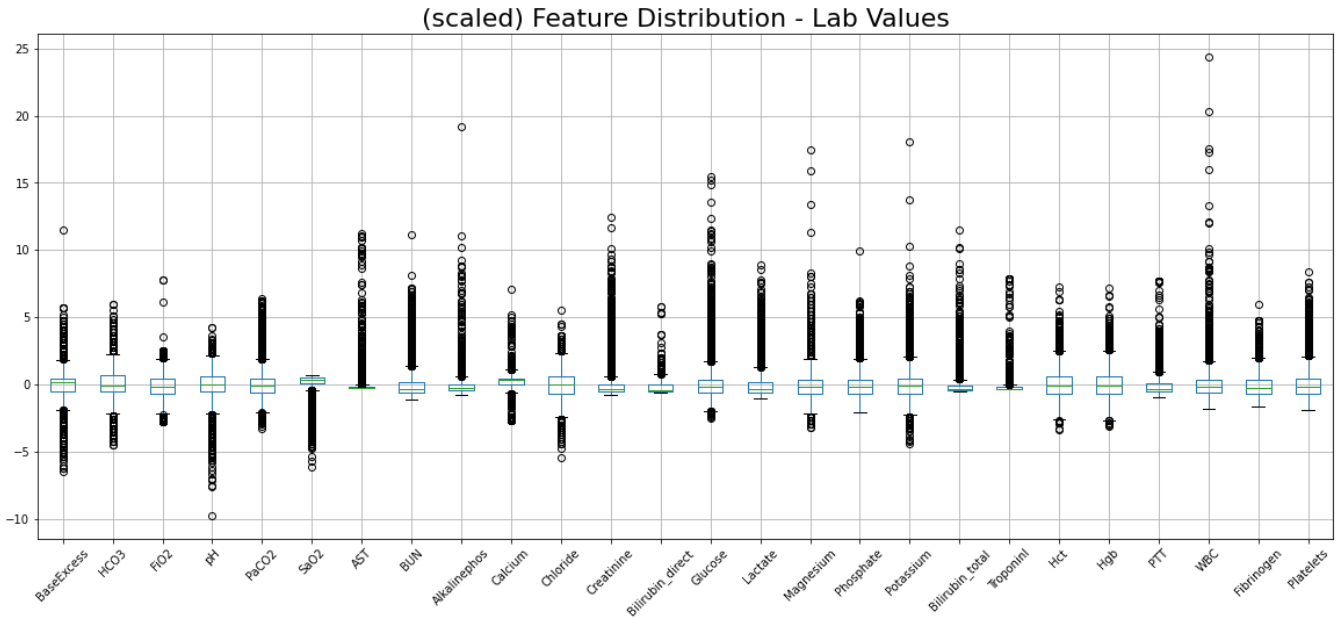
In order to visualize and compare distributions of all features, we first scale them, using standard scaling $\frac{x_i - \mu(x)}{\sigma(x)}$.

This scaling will help us next in the imputation part, and in the interpretability of linear models (weights are of the same order of magnitude).

```
StandardScaler()
```

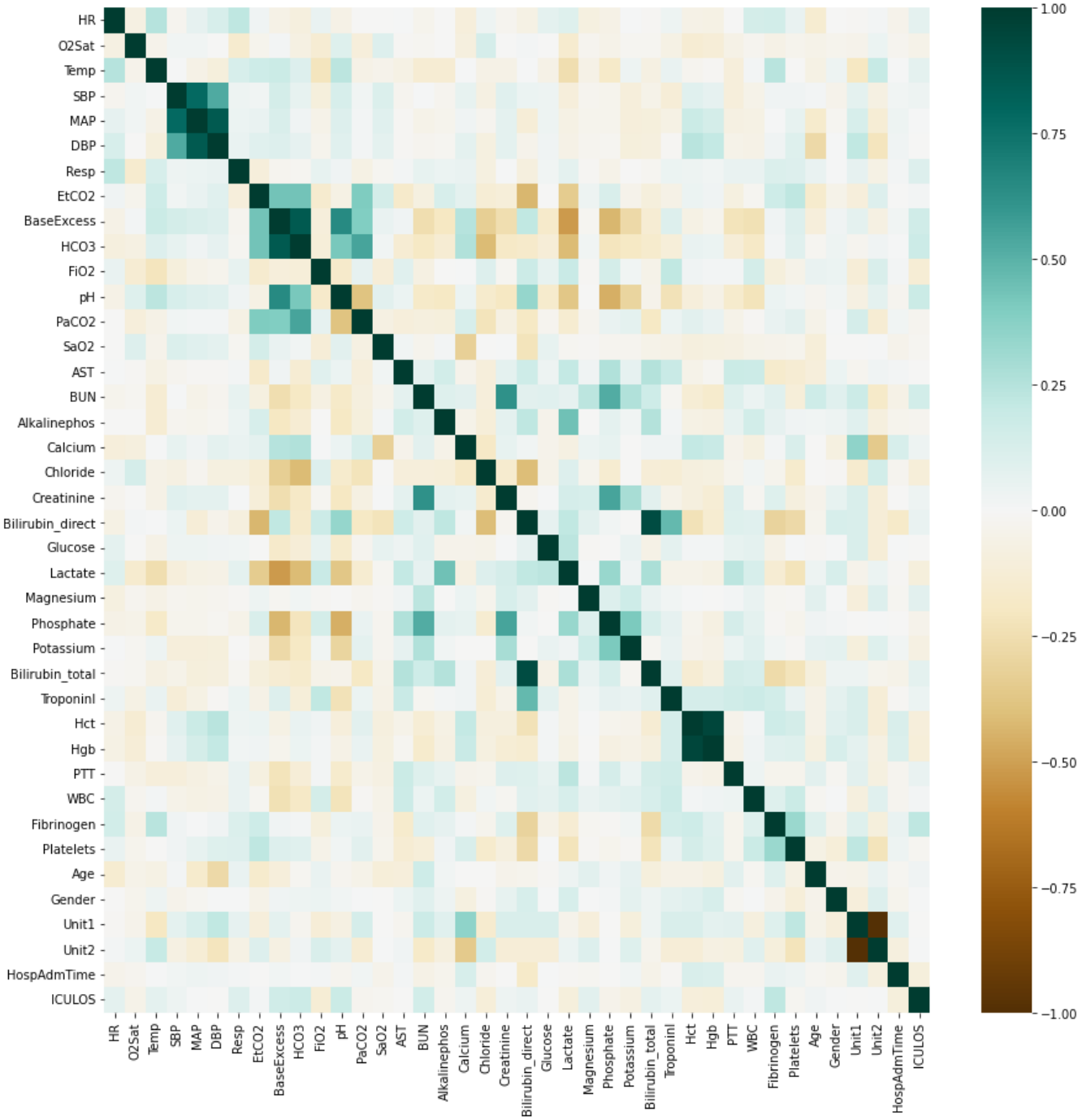
Now we can visualize all features distributions (vital signs and lab values) in the same plot.





Notice that most features have asymmetric distributions.

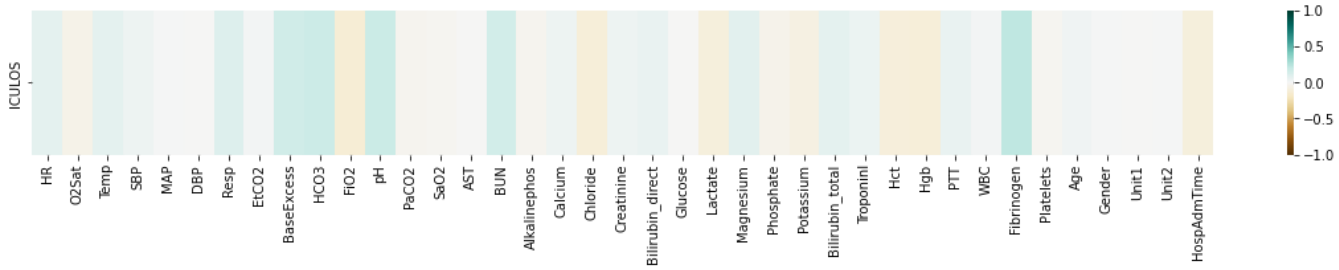
Pairwise Feature Correlations



We can observe that overall, features are not highly correlated with each other (for most i, j: $|\rho_{i,j}| < 0.25$).
Note that the correlation between 'Unit1' and 'Unit2' is -1, which means they encode the same information, therefore we will remove 'Unit2' in preprocess.

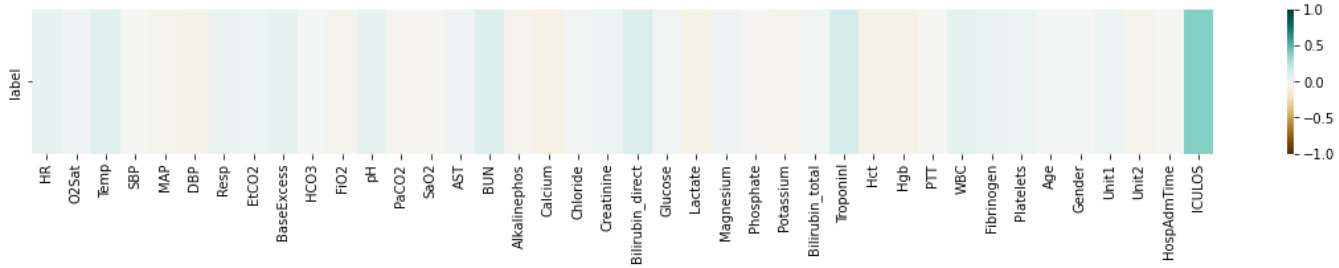
Correlation with time

We separately visualize the last row from the correlation matrix above, which includes the correaltion between ICULOS feature to other features. As ICULOS is the time-dimension, correlation with it is equivalent to correlation with time.



We can see that most vital signs features and demographic features are weakly correlated with time. While some lab values are correalted with time, notice that they appear rarely in the data (high precentage of missing values).
Time is important, and looking at a patient through time we can get more information. However, the above result implies that time is less significant in term of feature values. Due to the complexity of models that deal with timeseries data, and the result above, we choose to summarize and flatten the time dimension, as will be explained in the report (Feature Engineering section).

Correlation with target



We can see that most features have low correlation (linear relationship) with the target variable (SepsisLabel).
However, the ICU length-of-stay (ICULOS) is highly correlated with the sepsis label, therefore we expect it to be important in the sepsis prediction task (especially in a linear model).

Statistical Hypothesis Testing

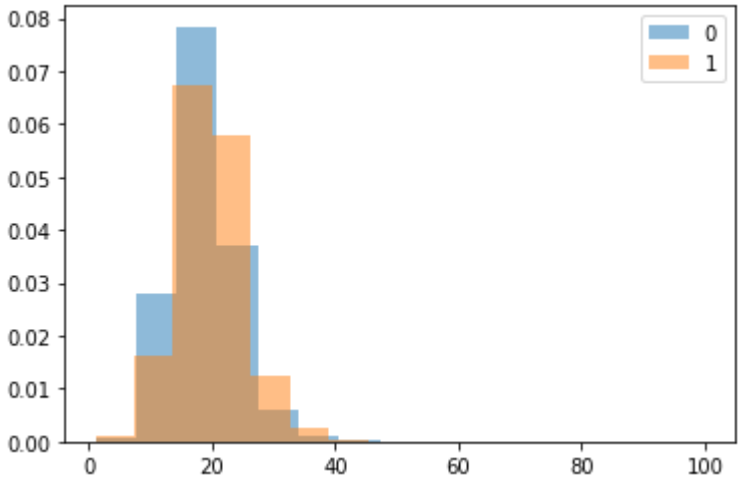
For each feature j we test whether its distribution is equal between the two groups (SepesisLabel=0 or SepesisLabel=1):
$H_0^j : F_0^j = F_1^j$
$H_1^j : F_0^j \succ F_1^j$ or $F_0^j \prec F_1^j$
where F_0^j and F_1^j are the CDFs for feature j for the two groups by sepsis label.

We decided to use a-parametric test (ranks sums) instead of t-test, since normallity assumption does not hold for most features. (P-values are adjusted using Bonferroni multiplicity adjustment)

```
Rejected hypotheses (adjusted p-values):
ICULOS, 0
HR, 7.87e-153
HospAdmTime, 1.03e-129
Temp, 2.41e-116
Resp, 1.73e-83
DBP, 3.15e-57
BUN, 7.8e-38
O2Sat, 1.18e-35
MAP, 6.67e-33
pH, 7.95e-22
WBC, 1.83e-20
Unit1, 1.79e-19
Unit2, 1.79e-19
BaseExcess, 1.84e-14
Calcium, 1.67e-10
Hct, 4e-10
Glucose, 1.67e-08
Creatinine, 7.7e-07
Hgb, 3.43e-06
Potassium, 8.9e-06
Age, 3.54e-05
EtCO2, 0.00257
Magnesium, 0.00479
TroponinI, 0.016
```

We can see that about half of the hypotheses were rejected, with relatively small p-values. The low p-values can be associated with the large number of samples, which enables to be identify small differences with high confidence.

For example, notice that the distribution of the feature 'Resp' for SepsisLabel=1 tends to larger values comapred to SepsisLabel=0:



Feature Engineering

Our main conclusions from the EDA which affect our feature engineering process are:

1. Vital signs features are rarely missing, while lab values features are frequently missing.
2. Most features are not correlated with time, hence time dimension can be flattened by statistical summarization.
3. The appearance of certain lab values is at least important as the value itself.

Based on these conclusions we have decided on an imputation policy and feature engineering, as will be described next:

Scaling

We have used standard scaling for all non-binary features. As we haven't used distance-based algorithms for prediction, this scaling is mainly useful for interpretability (e.g. comparison of linear model's weights is valid when features are in the same scale).

Data Imputation

We have decided on different policies of imputation (within each patient) depending on the feature group, as each group have different missingness properties:

1. Vital sign - imputation via linear interpolation, as most values are not missing, and therefore interpolation generalizes from close timesteps.
2. Lab values - imputation via interpolation will be limited in this case, as only a few timesteps are not missing for most patients. Therefore interpolation will typically depend on distant timesteps and might impute badly.
   We have decided to use backward/forward imputation which induces a step function based on existing values only. This technique has another advantage: it easily enables retrieval of the number of unique measures for each patient (which is considered by us as an important feature by itself).
3. Demographic features are usually not missing. Unit2 was removed (as it encodes the same information as Unit1). Unit1 was imputed by mode (as it is constant over time).
4. If a certain value was completely missing for a specific patient - we have imputed by 0 (the mean across all patients after standard scaling) or by drawing a random value from the distribution of all other patients.

Statistical Features

We use statistical features for each raw features in order to transform each patient representation from $T_i \cdot F$ matrix to $k \cdot F$-dimensional vector, where k is the number of statistical features.

In order to preserve as much information as possible after reducing the time dimension, we will use several kinds of statistical features, including measures of centrality and dispersion:

- Mean
- Median

- Standard Deviation
- Max of absolute values
- Min of absolute values
- Number of unique values (for stepwise-imputed feature this encodes the number of measurements)

We have also tried other statistics (other quantiles, value-range, IQR) which did not contribute significantly. Certain statistics were not taken for specific features as they did not make any sense. For instance, we did not take most statistical functions of Age as it is constant for each patient.

Implementation

While implementing the preprocess pipelines we have first created the labels for each patient (1 if there exists a timestep with SepsisLabel=1, 0 otherwise) and trimmed the irrelevant timesteps for patients with sepsis (all rows with SepsisLabel=1 except the first one). Then we have performed scaling, imputation and feature engineering as described above.

Prediction

Models and Hyperparameters Tuning

For the prediction task we have decided to evaluate the following learning algorithms:

1. Logistic Regression
    - We have used L2 regularization with c=1 (tuned as a hyper parameter).
2. Kernel SVM
    - We have tried different kernels: rbf (with different gamma parameter), and polynomial kernels of different degrees.
    - We have used L2 regularization with c=100 (tuned as a hyper parameter).
3. Random Forest
    - We have optimized the following hyper parameters: number of estimators, tree depths, number of features used by each tree, splitting criterion (entropy or gini). Notice that these hyper parameters are a form of regularization, and whenever we have observed overfitting (i.e., a large generalization gap) we have tried to reduce the trees' depth and number of features.
4. XGBoost
    - Same hyperparameter types as in Random Forest, and L2 regularization as well. We have searched over relatively large values of regularization constants as it is known that XGBoost tends to overfit.

We have used class weight to address the class imbalance problem whenever the models support these features (i.e., for all models except XGBoost). Hyper parameters were tuned by educated grid search for all models.

Results

The following table shows the best results obtained by each algorithms (in terms of F1 score on validation set), the optimal hyper parameters and various evaluation metrics on both datasets (train and validation):

model_name	train_f1	train_recall	train_precision	train_auc	test_f1	test_recall	test_precision	test_auc
log_reg	0.433	0.78	0.3	0.821	0.431	0.752	0.302	0.806
svm	0.647	0.866	0.516	0.902	0.503	0.66	0.406	0.791
random_forest	0.767	0.743	0.792	0.864	0.713	0.683	0.746	0.832
xgb	0.881	0.789	0.997	0.894	0.725	0.606	0.903	0.8

Final hyperparameters (rest is default): Logistic Regression (C=1); SVM (polynomial of degree 3 kernel, C=100); Random Forest (n_estimators=1000, 'entropy' criterion, max_depth=7, max_features=15); XGBoost (n_estimators=1000, colsample_bynode=10/n, max_depth=7, sampling_method='uniform', reg_lambda=1000).

The XGBoost model is the best in terms of validation F1-score, even though it is also the most overfitting model. Note that the logistic regression model is the worst in terms of F1-score, but it has obtained the best recall score on validation set. The gap between the random forest and the XGBoost seems insignificant in F1-score, but in terms of precision it seems that the XGBoost strictly outperforms the random forest model.

Interestingly, even though large differences can be seen in recall and precision, in terms of AUC all models obtain relatively similar results (random forest yields the best results). We consider the XGBoost as our best models (as the request of this assignment was to maximize F1-score), but notice that in many medical applications recall might have a larger weight in the general form of the F-beta score formula. This comes from the fact that type I error is costly in terms of money while type II error is costly in terms of life.

Model Interpretability

Attached next as a Jupyter notebook (view in GitHub):

Interpretability

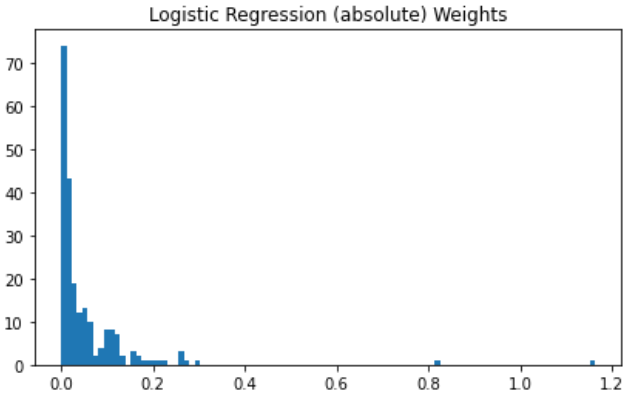
Besides predicting label for a new patient, the models we trained encodes additional information regarding feature importances.
Different models encodes this information in different forms, and we will focus on exploring the feature importances learned by Logistic Regression and Random Forest models.
Since Logistic Regression is a linear classifier, it will assign high importance to the features with strong linear relationship with the target, while Random Forest is a tree-based classifier, and we expect it to learn different (and potentially more complex) relationships.
In the following analysis we will adress the importance of features in different granularities:

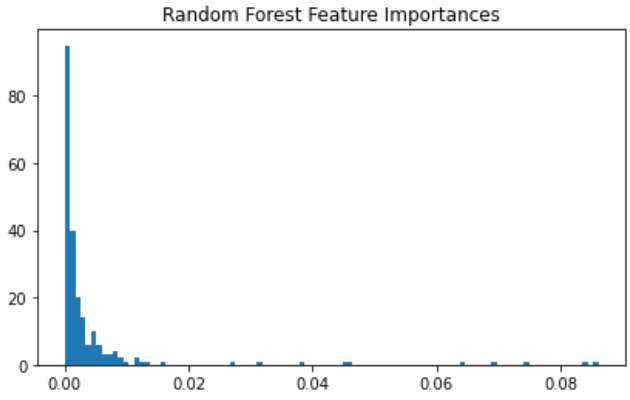
- Statistical features - statistic for each raw feature (acctual features used in the model)
- Raw features - features from original dataset (as aggregation of the statistical features importances)
- Feature-groups - groups of raw features (as aggregation of the statistical features importances)

Feature Importance Definition

For the Logistic Regression model, we define the importance of a feature by the absolute value of the corresponding coeffitient.
For the Random Forest model, the importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature.

Distribution of importances per model

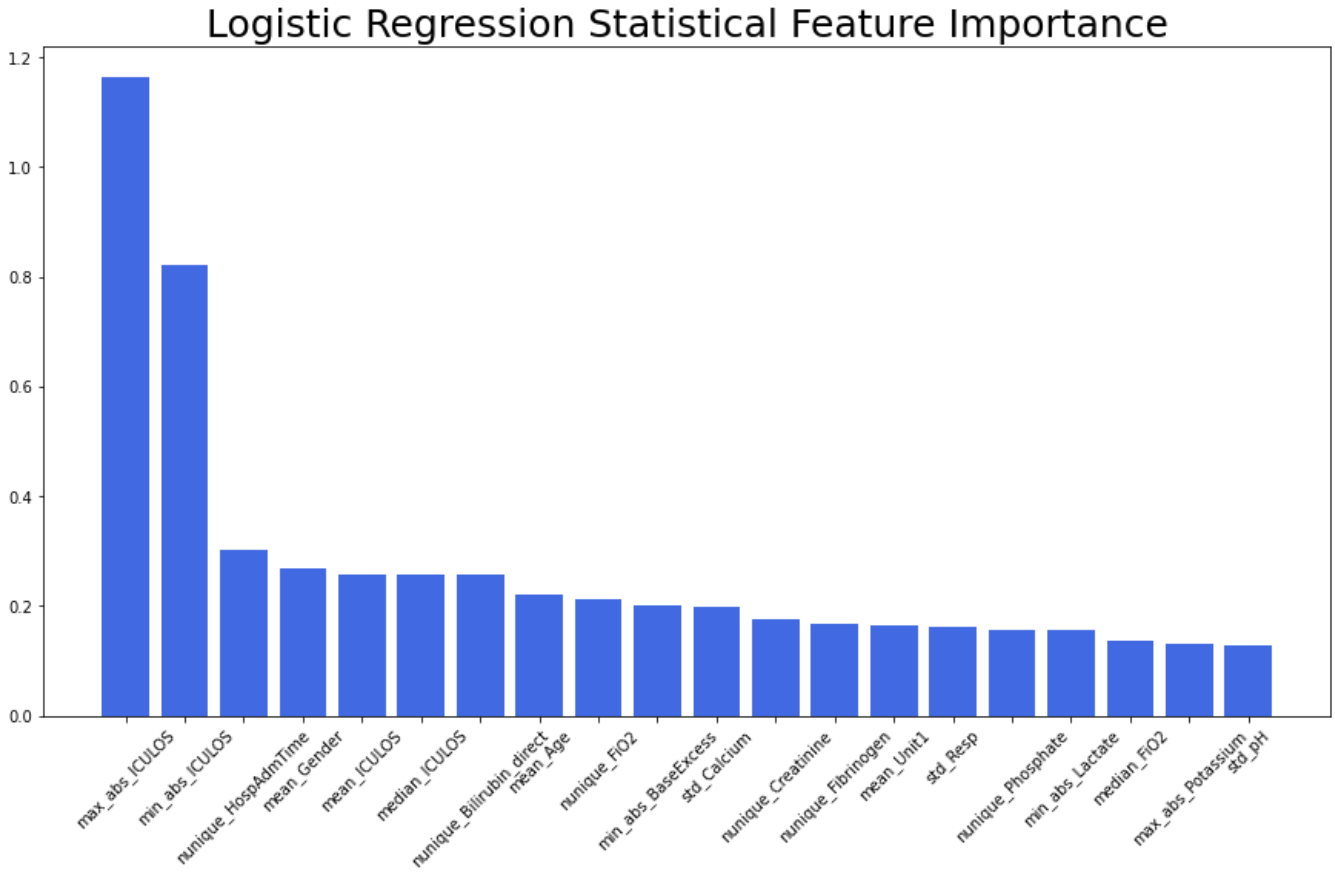


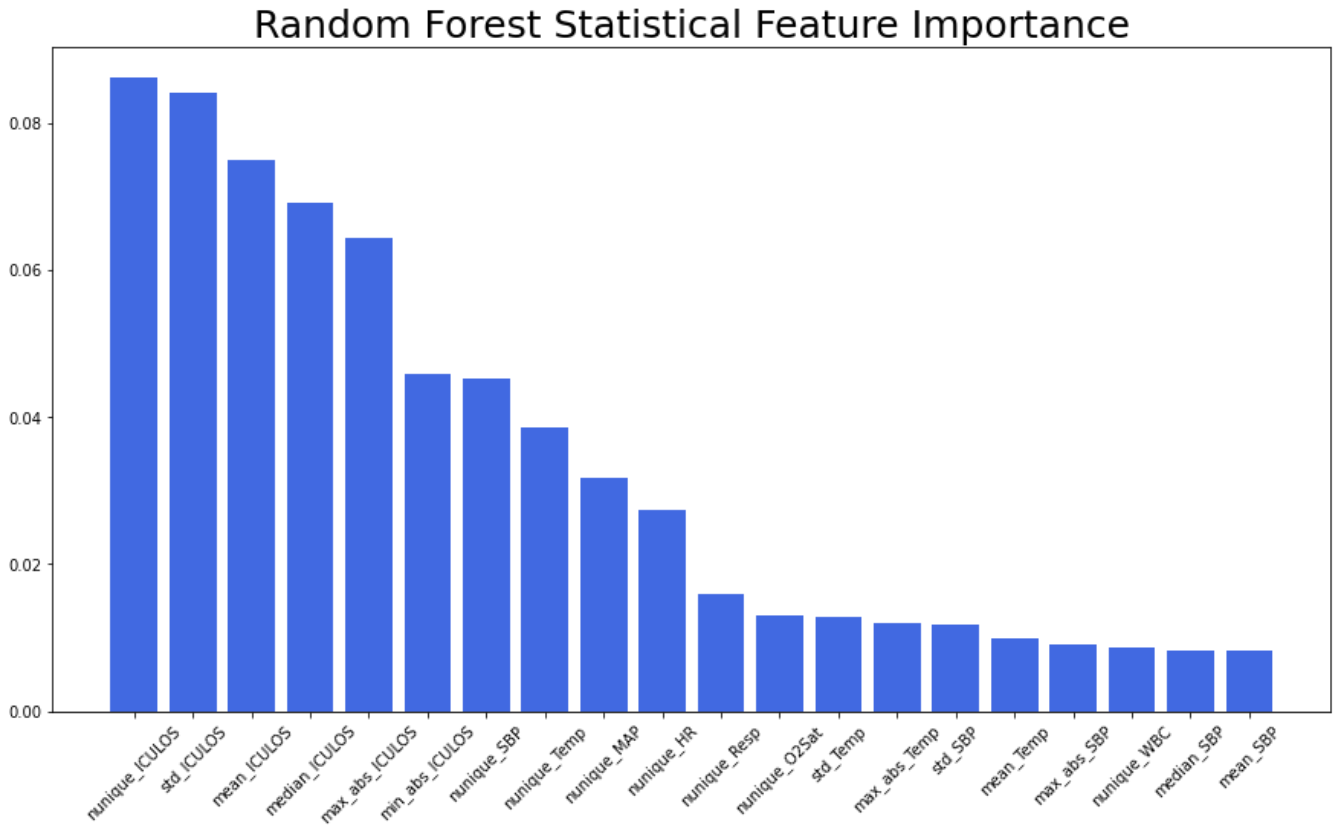


In both models, most statistical features have low importance (importance "score" close to 0), and only a few are significantly important.

Statistical Feature Importance

The following 2 plots visaulize the 20 most important statistical features used by each model (Logistic Regression, Random Forest).

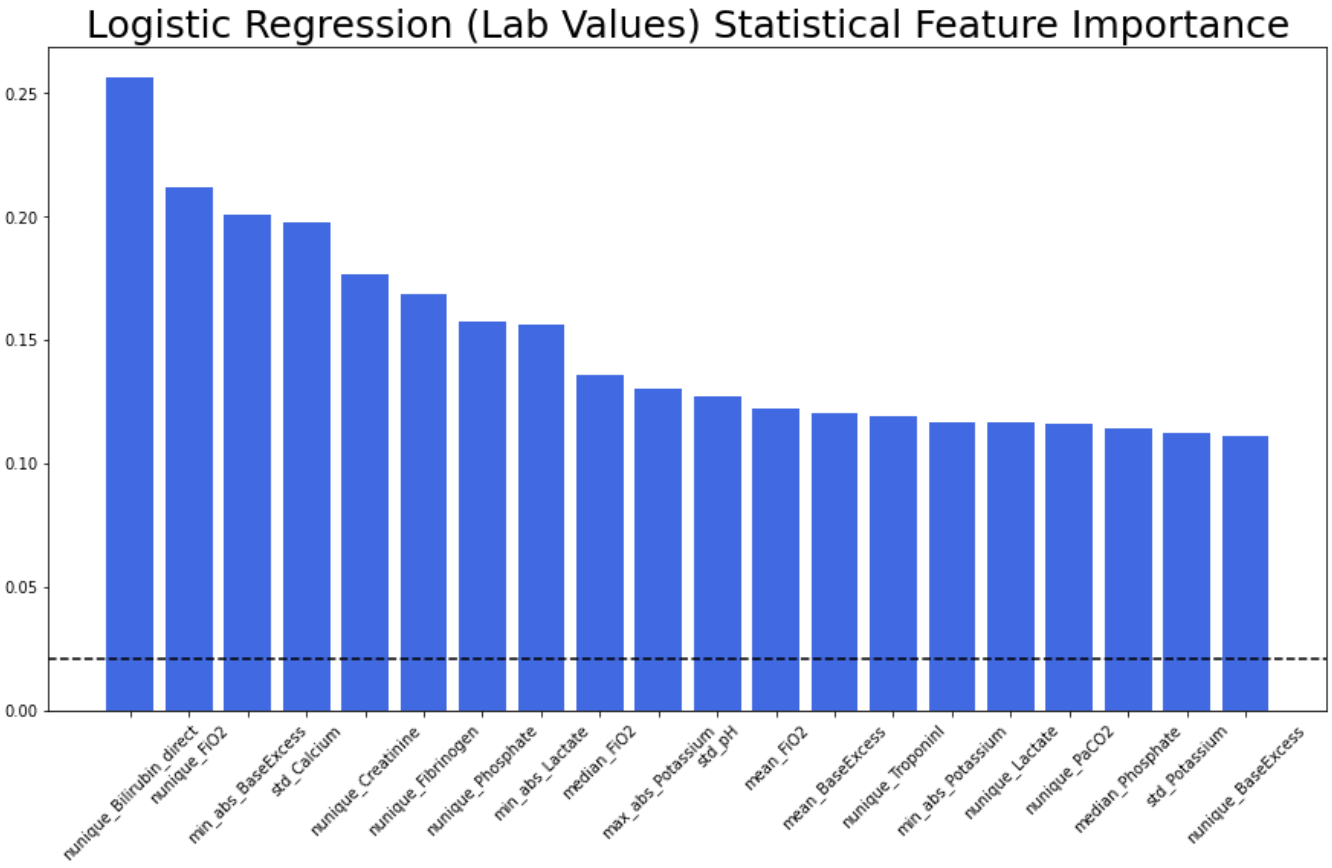


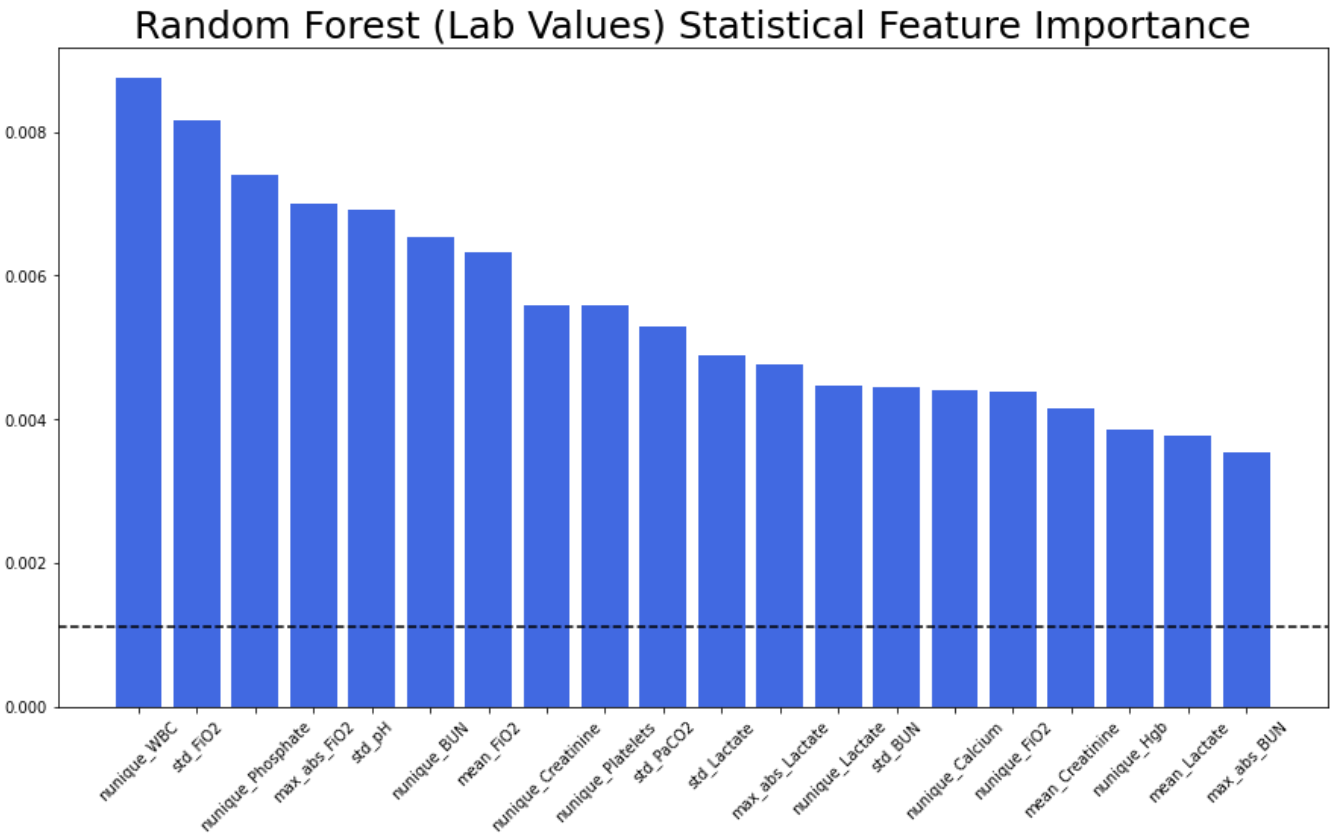


Note that both models modeled the feature 'ICULOS' as the most important feature. Notice that many important statistical features are similar to a "counter of timesteps", which indicates that number of timesteps is an important feature for both models.

Lab Values

Next, we will plot the statistical feature importance only for the lab-values feature group (dahsed line is the median importance over all statistical features):

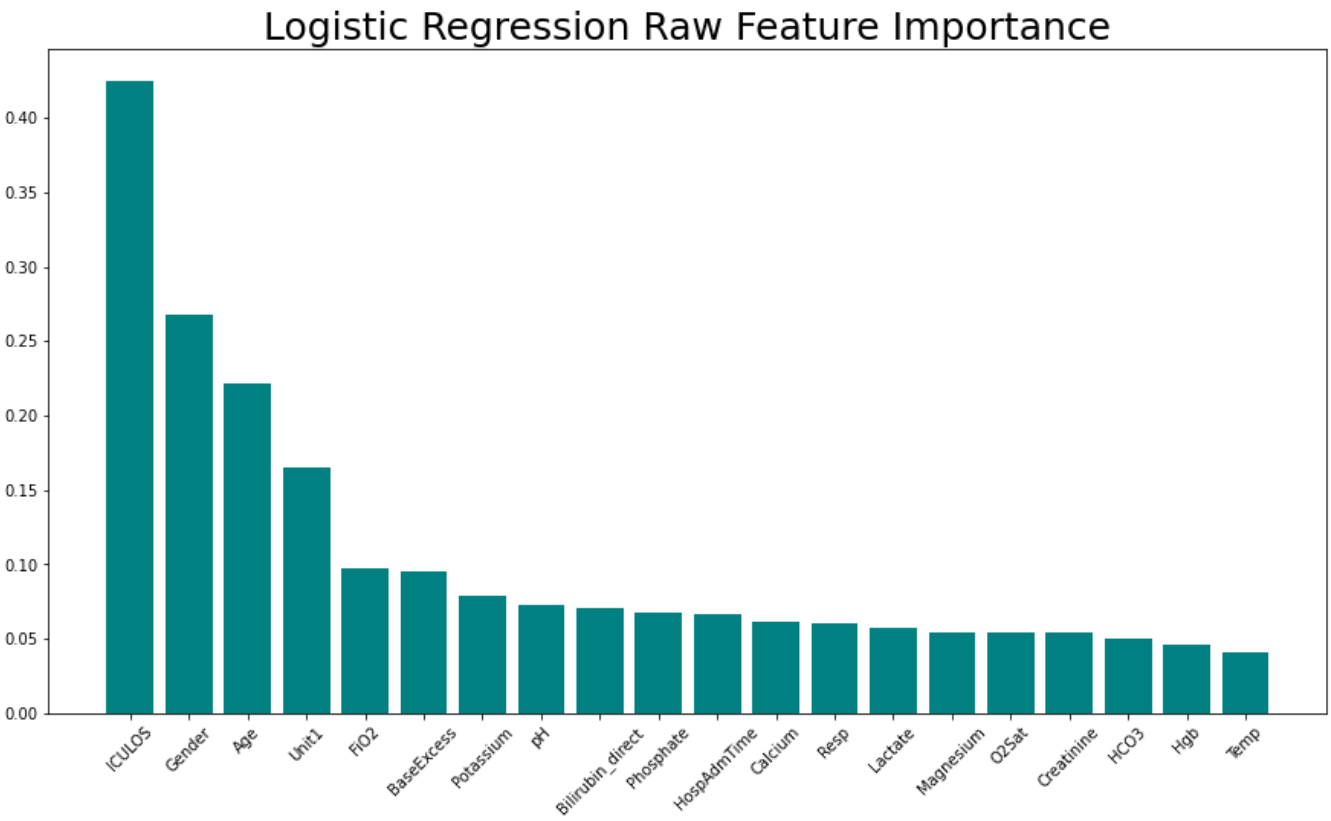


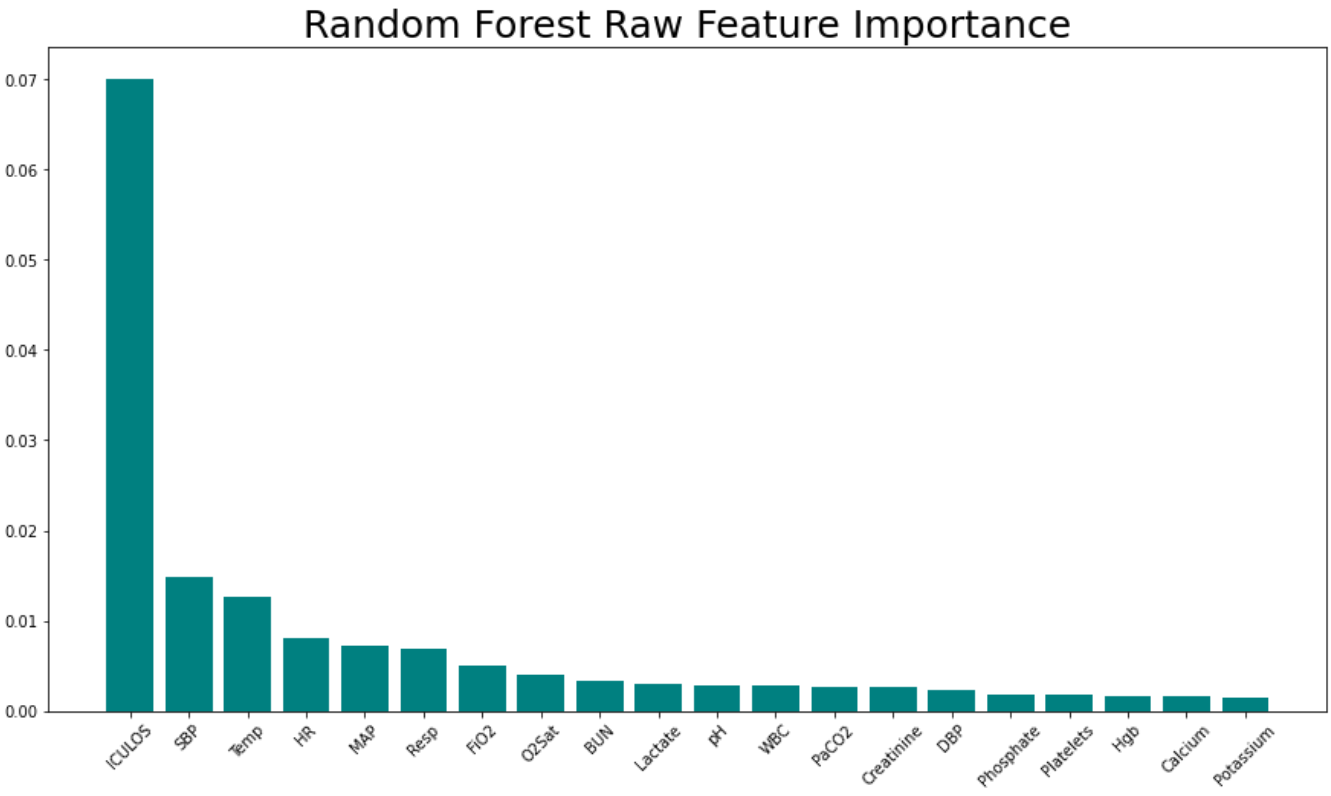


Notice that as we have expected (while decided on imputation policy and feature engineering), nunique is an important statistic on lab values features. This is because it encodes that a doctor found it valueable to perform this lab test.

Raw Features Importance

The following 2 plots aggregate the raw feature importance as the mean of all statistc-features importance associated with this raw feature.



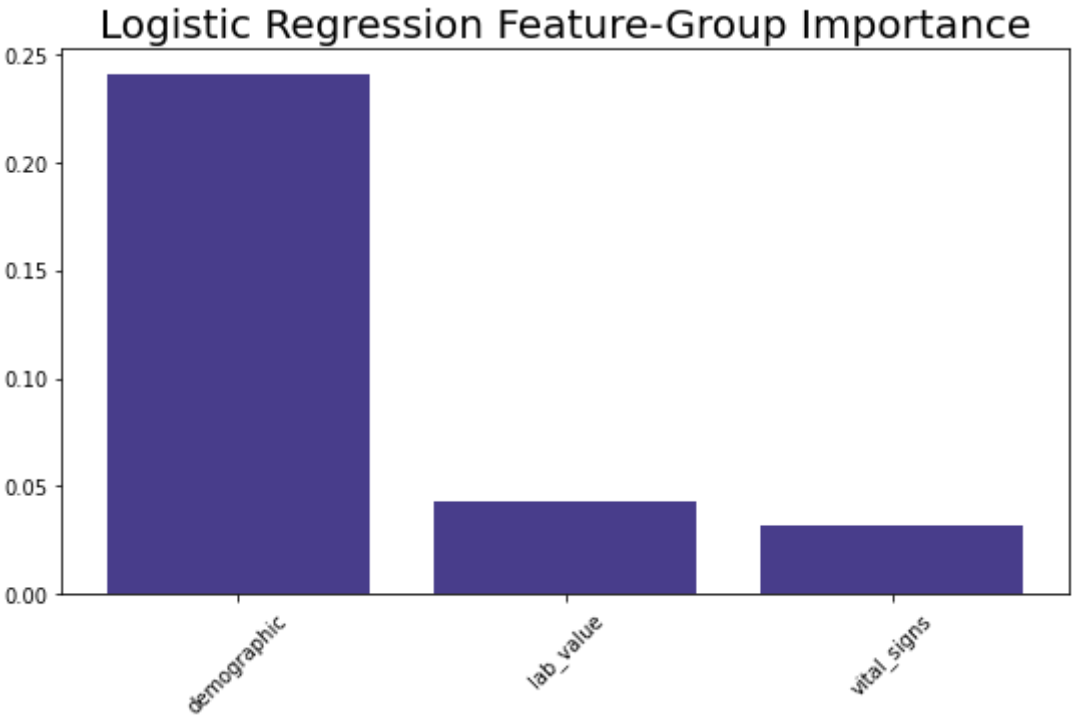


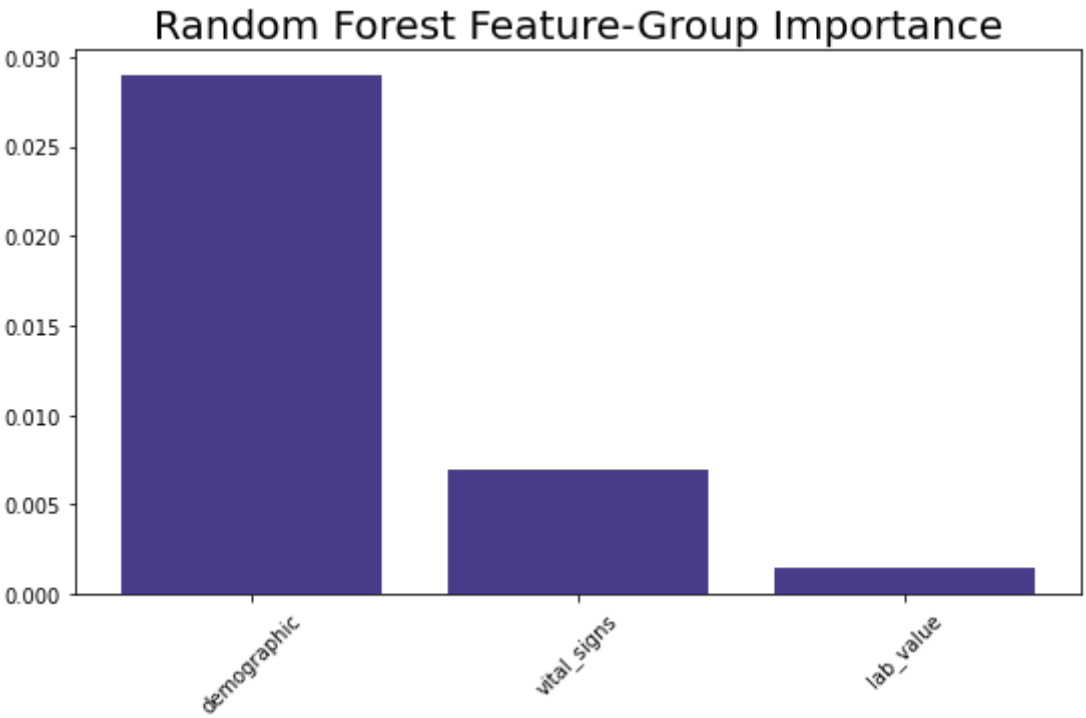
We can see a big difference between most important features between the models (except ICULOS). The Logistic Regression model assigns high weights to Gender and Age features, whereas Random Forest

mostly uses vital signs (and lab values) for prediction.

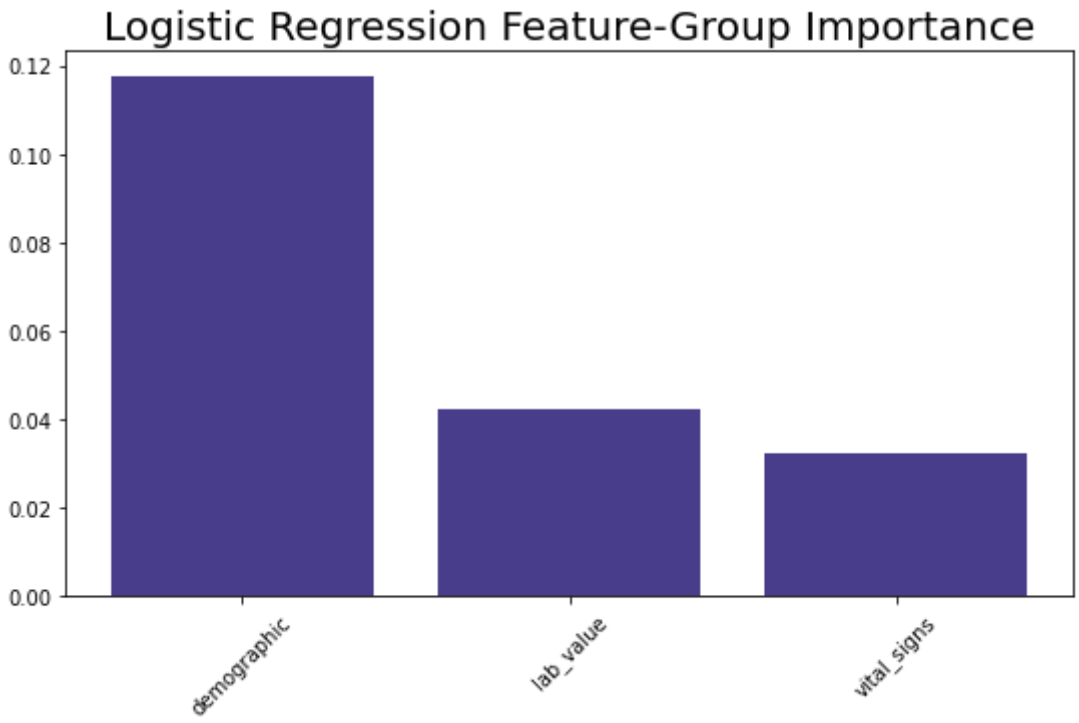
Feature-Group Importance

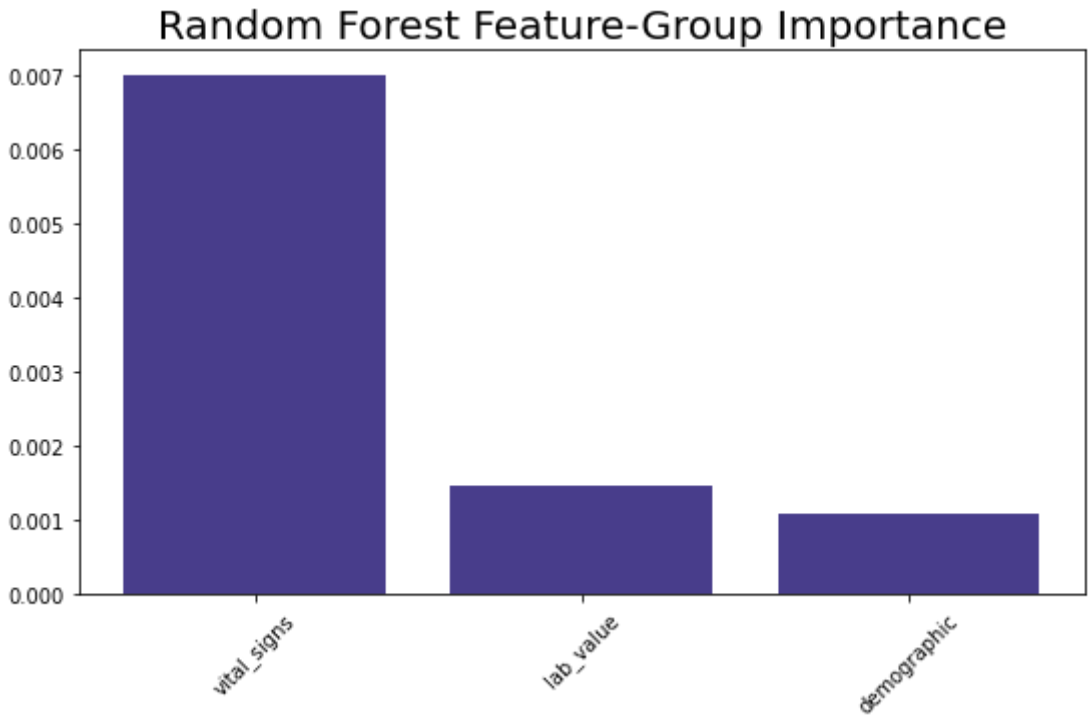
In the following 2 plots we visualize the aggregated importance of each feature group (vital signs, lab values, demographic)





The ICULOS feature is a part of the demographic feature-group, therefore it dominants the importance of this group.
Notice that when ingoring the ICULOS feature, a different picture is obtained:





While Logistic Regression still assigns more weight to demographic features (e.g. Age, Gender), in Random Forest, we see that vital signs are more dominant. Moreover, demographic (excluding ICULOS) are less important than lab values features (for Random Forest).

Summary and Discussion

Clinical data and prediction tasks involve various challenges: time dependent data, frequently missing data, class imbalance and different types of data (some features are time-dependent and others are not). Identifying these challenges early on in the EDA phase proved to be useful as it enabled us to design proper imputation and feature engineering policies.

We have tried to define a more concise representation of a patient, in a way that preserves as much information as possible, and enables using relatively simple (yet powerful) learning algorithms.

In this specific prediction task the tree-based methods performed better than the linear methods in terms of F1-score. One possible explanation could be the fact that decision trees might model more complex relationships between the features and the target (e.g. interactions), and they are similar to a way that a human doctor makes a decision given these features. For example, a certain lab value is important only in the case where another lab value is too high - these kinds of rules are exactly the ones captured by decision trees.