HW1 - Early Prediction of Sepsis from Clinical Data

1. Executive summary:

- Data analysis and exploration – finding relevant features by analyzing the distribution of the features and counting the missing values for each, and by gaining a deeper understanding of what sepsis is from a medical perspective and how it's diagnosed.
- Feature engineering – aggregating the features in several techniques and using hypothesis tests to determine which features will have more impact. The best performing technique was to take the mean of the last 5 rows for each patient on all features.
- Creative aggregation – weighted average –assuming that information closer to the diagnosis is more relevant than past information, we test a weighted mean on all of the features $\sum_{i=n}^{1}(sample_i) * \frac{1}{2^i}$, where n is the latest sample for a specific patient. While this technique showed strong results, it was outperformed by a regular mean.
- Missing data imputation – During the experiments, we tested multiple data imputation techniques such as Mean, Median and KNN imputation. Ultimately the selected model requires no data imputation, so no imputation was performed.
- Model training – Multiple models were tested, such as XGBoost, Random-Forest, and Ada-boost to predict Sepsis in patients. After hyper-parameter tuning the best performing model was XG-boost. All models were evaluated on their F1 performance. The train and test F1 scores were compared and metrics that indicated that our model was overfitter wasset to avoid overfitting.
- Post analysis – We took additional measurements and discovered that the Random forest model is likely to overfit compared to the other models. We found that the XGboost model achieves better performance for women than men and that the feature importance of the model was reasonable and explainable.

2. Exploratory Data Analysis:

a. Describing the features that are available in the dataset:

The dataset we are working with contains a total of 41 features. These features include demographic information, vital signs, laboratory results, and other clinical information about patients who have been admitted to the ICU. Our team decided to begin by conducting thorough research on each of the 41 features to gain a better understanding of their individual roles and relevance to our project. We discovered that Sepsis is a potential complication that can arise in patients who have been admitted to the ICU.

The features that are available in the dataset are:

- HR: Heart rate in beats per minute
- O2Sat: Oxygen saturation level as a percentage

- Temp: Body temperature in degrees Celsius
- SBP: Systolic blood pressure in mmHg
- MAP: Mean arterial pressure in mmHg
- DBP: Diastolic blood pressure in mmHg
- Resp: Respiratory rate in breaths per minute
- EtCO2: End-tidal carbon dioxide level in mmHg
- BaseExcess: Base excess level in blood, measured in mmol/L
- HCO3: Bicarbonate level in blood, measured in mmol/L
- FiO2: Fraction of inspired oxygen, expressed as a percentage
- pH: Blood pH level
- PaCO2: Partial pressure of carbon dioxide in arterial blood, measured in mmHg
- SaO2: Arterial oxygen saturation level, expressed as a percentage
- AST: Aspartate aminotransferase level in blood, measured in U/L
- BUN: Blood urea nitrogen level in mg/dL
- Alkalinephos: Alkaline phosphatase level in blood, measured in U/L
- Calcium: Blood calcium level in mg/dL
- Chloride: Blood chloride level in mmol/L
- Creatinine: Blood creatinine level in mg/dL
- Bilirubin_direct: Direct bilirubin level in blood, measured in mg/dL
- Glucose: Blood glucose level in mg/dL
- Lactate: Blood lactate level in mmol/L
- Magnesium: Blood magnesium level in mg/dL
- Phosphate: Blood phosphate level in mg/dL
- Potassium: Blood potassium level in mmol/L
- Bilirubin_total: Total bilirubin level in blood, measured in mg/dL
- TroponinI: Troponin I level in blood, measured in ng/mL
- Hct: Hematocrit level, expressed as a percentage
- Hgb: Hemoglobin level in g/dL
- PTT: Partial thromboplastin time in seconds
- WBC: White blood cell count in cells/mm3
- Fibrinogen: Fibrinogen level in blood, measured in mg/dL
- Platelets: Platelet count in cells/mm3
- Age: Age of the patient in years
- Gender: Gender of the patient (0 for male, 1 for female)
- Unit1: Indicator variable for whether the patient was assigned to unit 1 (0 for no, 1 for yes)
- Unit2: Indicator variable for whether the patient was assigned to unit 2 (0 for no, 1 for yes)
- HospAdmTime: Time elapsed between hospital admission and ICU admission, measured in hours
- ICULOS: Number of hours that the patient has been in the ICU at the time of measurement
- SepsisLabel: Binary indicator variable for whether the patient has been diagnosed with sepsis (0 for no, 1 for yes)

b: Inspecting the features distribution, comparative analysis:

After conducting initial research and observing many features with varying degrees of missing data, our primary goal was to determine the significance of each feature and to understand which ones were most relevant. Through this process, we aimed to gain a deeper understanding of the dataset and identify key features that could be used later.

Our analysis revealed that patients who developed Sepsis had spent significantly more time in the ICU compared to those who did not: <maybe add graphs>

We can also observe the heavy tail of the distribution. This is an interesting observation, since as far as we know sepsis is a severe condition that gets worse in a few hours, and the distribution shows that there is a significant number of patients that spent over 50 hours before they were diagnosed, meaning those patients got sick with sepsis while they were in ICU.

Additionally, we decided to explore the number of NaN values in each feature to identify the features with the highest number of missing values and decide on whether to drop them, impute the missing values, or select an alternative feature. This can help to improve the quality and effectiveness of the machine learning model by ensuring that we don't use features with too many NaN values (that might be completed not perfectly) and add noise to our prediction.

	Column	Average Non-Null Values
0	HR	0.900935
1	O2Sat	0.869339
2	Temp	0.339227
3	SBP	0.852870
4	MAP	0.875071
5	DBP	0.688105
6	Resp	0.845686
7	EtCO2	0.035782
8	BaseExcess	0.054866
9	HCO3	0.041478
10	FiO2	0.083190
11	pH	0.070094
12	PaCO2	0.056078
13	SaO2	0.035001
14	AST	0.015963
15	BUN	0.068507
16	Alkalinephos	0.015795
17	Calcium	0.058571
18	Chloride	0.045093
19	Creatinine	0.060769
20	Bilirubin_direct	0.001893
21	Glucose	0.171751
22	Lactate	0.026789
23	Magnesium	0.062887
24	Phosphate	0.039718
25	Potassium	0.093400
26	Bilirubin_total	0.014711
27	TroponinI	0.009683
28	Hct	0.088837
29	Hgb	0.073976
30	PTT	0.029073
31	WBC	0.063972
32	Fibrinogen	0.006421
33	Platelets	0.059266
34	Age	1.000000
35	Gender	1.000000
36	Unit1	0.609899
37	Unit2	0.609899
38	HospAdmTime	1.000000
39	ICULOS	1.000000
40	SepsisLabel	1.000000

We observed that the dataset contains numerous missing values, with varying percentages of missing rows across the features.

Our first step was to only include features that appear at least once in at least 90% of patients. We decided to examine only those features in further analysis since we believed the others are

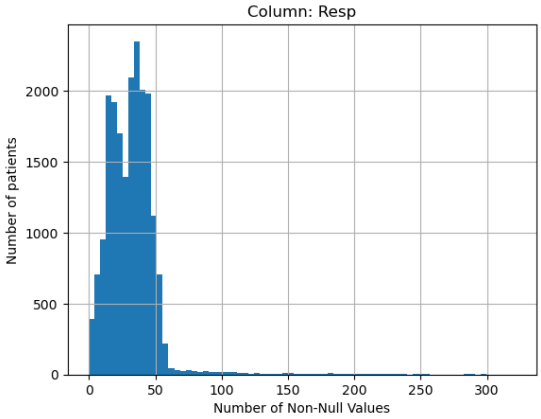
too sparse for the model to learn. Then, by examining plots that display the number of missing values for each feature, we identified three main categories of features:

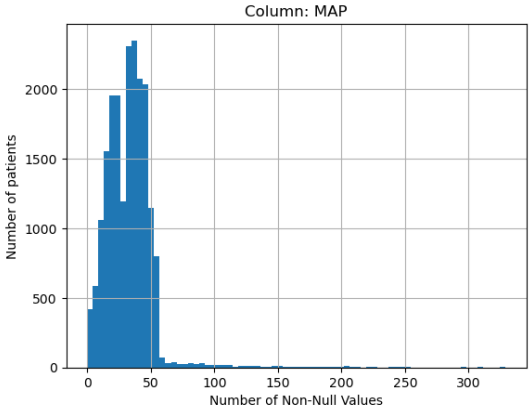
`Hourly features` collected hourly or every few hours, such as heart rate, temperature, and time in ICE

`Rare features` collected through blood tests and resulting in sparse data with null values since they are not performed every hour.

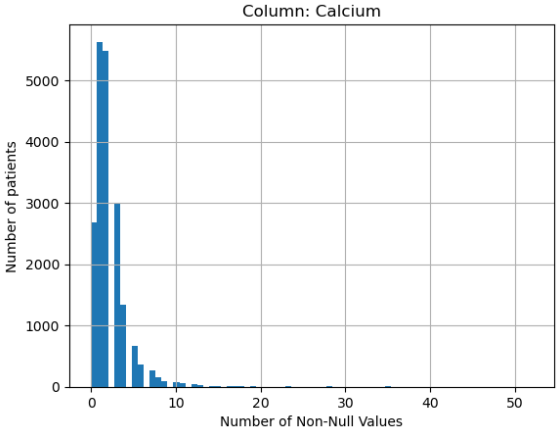
`Target feature` , which is the label we aim to predict.

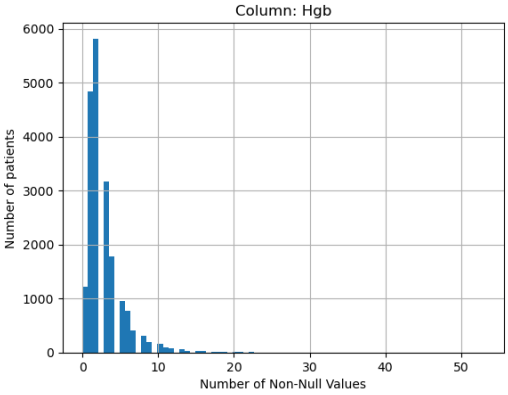
Hourly features examples:





Rare features examples:





When comparing the ‘hourly features' to the ‘rare features’ we can observe that the number of non-null values is significantly higher for the ‘hourly features’. This is reasonable, as these measurements are automatically or more easily obtained on an hourly basis compared to 'rare features', which may not be checked as frequently (blood tests, for example). Since our preliminary research suggested that the rare features are necessary to detect sepsis, we decided to use the rare features even though they are sparse. Upon analysis, we noticed that for the rare features, a significant number of patients had less than 10 non-missing entries available. Therefore, we tried to treat each group differently in the feature engineering and imputation phases.

c. Handling missing data:

As we aggregate the series data (we will elaborate on the aggregation in the pre-processing phase), we decided to handle the missing data by completing the missing values with the average of all patients in each feature. Completing missing data with the average of all patients in a feature allows us to retain as much information as possible while still filling in the gaps in the data. It also ensures that the completed data points are consistent with the other values in the same feature, which is important for maintaining the integrity of the dataset.

Features Engineering:

a. Which feature you will be using (and why)

We tried two approaches:

1. Take only the features with 90% of non-null values among patients.
2. Take all features.

For the XGBoost and Random-Forest models, we used the 'Embedded Feature Selection technique' as well, for both types of feature groups. Embedded Feature Selection is a technique used in machine learning to automatically select the most relevant features for a given model during the model training process. This approach involves integrating feature selection into the process of building the model itself, rather than performing feature selection as a separate preprocessing step.

In our case, we used embedded feature selection because we had many features, some of which were very technical (and we might not fully understand their meaning). By incorporating feature selection directly into the model training process, we hoped to improve the performance of our model and reduce the risk of overfitting. This is also the reason we trained the model on all features – we let the model choose the most informative features.

b. Features transformations:

We tried two different approaches for transformation:

1. Different transformations for each feature group – calculating the mean, std, median, minimum/maximum value, and other quantiles of the 'hourly features' and adding them as additional features. This was done because we believed that the statistics of the patients might be just as important as the regular values. If the measurements are changing a lot or not, it might indicate a medical problem, so we decided to add these features and check if they are relevant. In addition, calculating the minimum and maximum values can provide insights into the range of values observed for each feature, which can be useful for identifying outliers or anomalies (which can be considered by the model).

Overall, these statistical features can help provide a more comprehensive understanding of the patient's health and help identify patterns and trends that may be indicative of illness.

The transformed features (mean, median, std, Q1, Q3, minimum value, maximum value) were used only if they displayed a statistically significant difference between sick and non-sick patients, based on T-Test analysis. For example ($\alpha = 0.05$):

```
Column = SBP_min
Reject null hypothesis. The two classes are significantly different.
P-value = 0.0004315461031596228

Column = SBP_50%
Reject null hypothesis. The two classes are significantly different.
P-value = 0.0004376297915353784

Column = SBP_max
Fail to reject null hypothesis. The two classes are not significantly different.
P-value = 0.4019725756672471

Column = MAP_mean
Reject null hypothesis. The two classes are significantly different.
P-value = 8.788147784134939e-09
```

For the rare features, we only took the weighted average – we believed that using classic statistical measures will be less informative for the model since there are not enough observations to get close to the real distribution of the features.

To account for the fact that time plays a crucial role in predicting the onset of sepsis, we employed a **weighted average** on the different rows of our time series data. **We believed that the information contained in each row becomes increasingly relevant as the diagnosis approaches**. Therefore, we assigned higher weights to the rows that are closer to the diagnosis and lower weights to those further away. This allowed our model to capture the patterns and signals that are most indicative of sepsis and improve its predictive accuracy.

2. **We took the mean for all features without differentiating hourly and rare features -** We tried to take the mean for each column for each patient. We tried averaging over different time- periods:
   - over the whole period
   - over the last 10 hours before diagnosis
   - over the last 5 hours before diagnosis
   - taking the last row before diagnosis

We believe that taking the last rows will give us more informative information about the patients, as we know sepsis progresses quickly and those are the more critical hours for detection.

<u>Feature Enrichment - Using post-6-hours data</u>

During our data exploration and research on the subject, we identified an issue with the dataset - the label we are trying to predict is the time of diagnosis, which can vary greatly across patients due to factors such as medical attention and prior conditions. **We hypothesized that some patients may have been diagnosed at different stages of the disease**, leading **to overlap in their medical information**. To address this issue and improve our model's predictive power, we

decided to include measurements of sick patients that were taken after the 6-hour mark prior to diagnosis. Although these additional rows won't be included in our test set, we believe they can provide inductive value and help the model learn which signals are associated with the disease.

4. Prediction

In the modeling phase, we experimented with 3 different algorithms, XGBoost, AdaBoost, and Random Forest. All the selected algorithms are tree-based, and as previously mentioned all these models have feature selection embedded in the algorithm.
After training with the different approaches we presented for handling the missing data, for the feature selections (drop sparse columns, remove features without statistical significance based on t-tests) and for feature enrichments (taking weighted means, adding rows from future and taking statistical measures) , we found **that taking only the last 5 rows for each patient and averaging all the features in the dataset per patient and impute them with the average among all patients is the best input for our models** (achieved the best results for all 3 models).

1. Hyperparameter selection (for all models):

To find the best hyperparameters for each model, we used the Optuna module for an effective search of optimal hyperparameters. (https://arxiv.org/pdf/1907.10902.pdf)
Optuna is a hyperparameter optimization framework. It provides an automated and efficient way to search for the best set of hyperparameters for a given machine-learning model.
Optuna uses Bayesian optimization to intelligently explore the hyperparameter search space. It begins by randomly selecting a set of hyperparameters and then evaluates the model's performance with those hyperparameters. Based on the performance, it constructs a probability distribution of the hyperparameters, which is used to sample new sets of hyperparameters for evaluation. This process is repeated for a predefined number of trials, and the best set of hyperparameters is selected based on a predefined performance metric (in our case -F1).

XGboost:

Train F1: 0.81 , Test F1: 0.74

XGBoost is a popular gradient-boosting framework that is designed to handle large and complex datasets. It uses a combination of decision trees to model the data and gradient boosting to improve the accuracy of the model. One of the benefits of XGBoost is that it can handle missing values in the dataset. By default, XGBoost treats missing values as zeros, but it can also learn how to handle missing values by imputing them with the most frequent value or a user-defined value during the model training process. Additionally, XGBoost can automatically learn how to

best deal with missing values by splitting the data based on the presence or absence of a feature value.

Best hyperparameters for the model - max_depth': 6, 'learning_rate': 0.020021827800040026, 'n_estimators': 409, 'min_child_weight': 8, 'gamma': 0.002376046358768467, 'subsample': 0.3267777415133442, 'colsample_bytree': 0.9922434396349483, 'reg_alpha': 0.03345778334794334, 'reg_lambda': 0.00023611208593944376, 'scale_pos_weight': 2

Random Forest:

Train F1: 0.85, Test F1: 0.68

Random forest is an ensemble learning method that constructs a multitude of decision trees at training time and outputs the mode of the predictions of individual trees. The key idea behind random forest is to introduce randomness into the tree-building process to avoid overfitting and improve generalization performance.

ADAboost:

Train F1: 0.66, Test F1: 0.66

Random forest is an ensemble learning method that constructs a multitude of decision trees at training time and outputs the mode of the predictions of individual trees. The key idea behind random forest is to introduce randomness into the tree-building process to avoid overfitting and improve generalization performance.

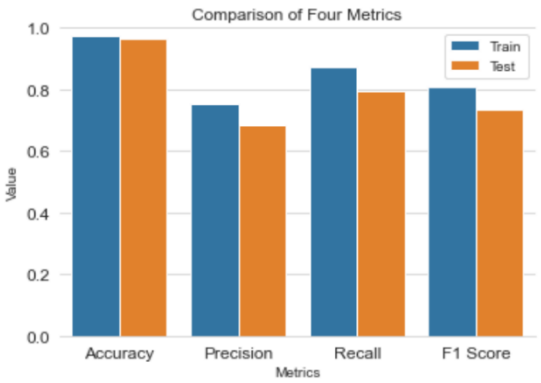
We can see that XGboost achieved the best results.

Post Analysis:

We calculated some metrics for each model:

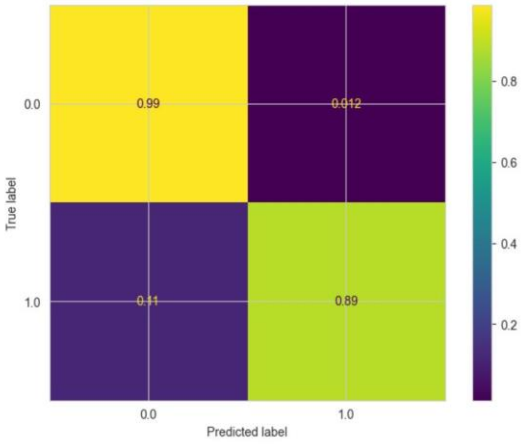
XGboost:

	Accuracy	Precision	Recall	F1
Train	0.97	0.75	0.87	0.81
Test	0.96	0.68	0.8	0.74

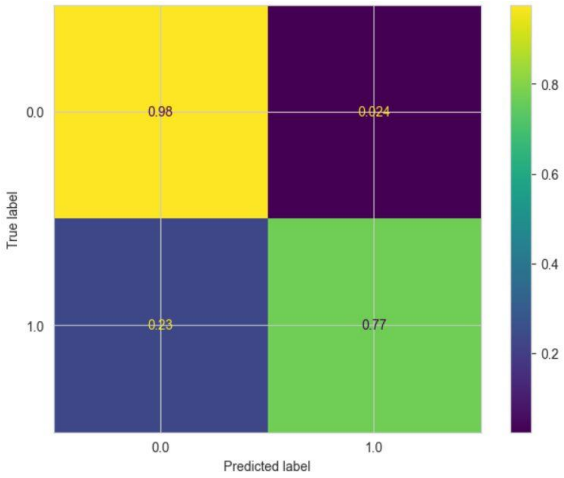


We can see that all measures are relatively closen when comparing the train set to the test set. We can see that the Recall measure is quiet high which means the model successfully identified Sepsis in patients that had Sepsis.

Confusion matrix for train dataset:



confusion matrix for test dataset:



We also compared the F1 scores between different groups:

	Women F1	Men F1
Train	0.8005	0.816
Test	0.7424	0.7259

We can see that in the train set the results are quite similar but in the test the F1 score is higher for women. Th is might suggest that the model will perform better for dataset with more women than men.

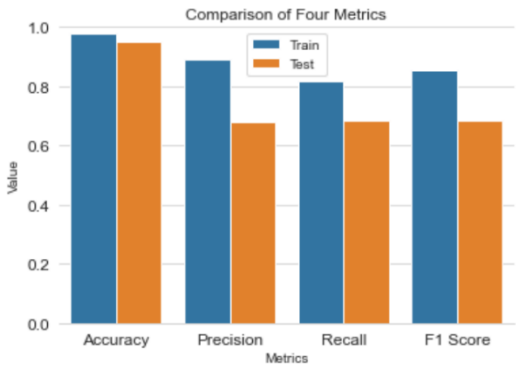
	Adults (18-55)	Elderly (above 55)
Train	0.822	0.8002
Test	0.707	0.7489

We can see that the model was able to generalize better for the elderly group since the difference between the train and test results is lower for this group.

Random Forest:

	Accuracy	Precision	Recall	F1
Train	0.98	0.89	0.82	0.85
Test	0.95	0.68	0.69	0.68

For all measures, the train results are much higher than the test results which is a strong indicator for overfit.



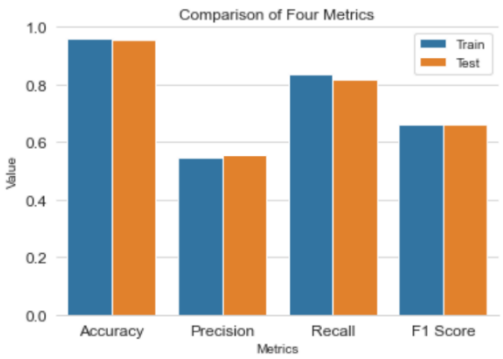
	Women F1	Men F1
Train	0.84	0.86
Test	0.68	0.67

	Adults (18-55)	Elderly (above 55)
Train	0.85	0.85
Test	0.65	0.69

We can see that even when analyzing different sub-groups, the differences between the train and test are large – meaning the model overfitted even for the sub-groups.

Ada- Boost:

	Accuracy	Precision	Recall	F1
Train	0.96	0.55	0.83	0.66
Test	0.96	0.55	0.82	0.66



We can see that all measures are close when comparing the train to the test, which means that the model was able to generalize properly, but the relatively low F1 score indicates that the model was not able to capture the complex structure of the data.

	Women F1	Men F1
Train	0.662	0.65
Test	0.666	0.652

	Adults (18-55)	Elderly (above 55)
Train	0.669	0.65
Test	0.64	0.66

We can see that the model did not overfit and the results for the sub-groups are similar to the results for the group of all patients.

Model interpretability:

XGBoost provides a way to interpret the feature importance of the model. We used the 'plot_importance' function to visualize the top 10 most important features:

We can see that the features we expected would affect the model such as body temperature or heart rate were marked as important features by the model. Additionally, we were not surprised to find that the ICULUS feature (time in ICU) had such a significant impact on the model. In our preliminary analysis we found that patients that were diagnosed with sepsis spent a lot more time in ICU than the others in the hours before diagnosis.