Data Mining Project Report

Predicting Hospital Mortality

Pascal Wyler & Rayhan Aurelio

MS in Applied Data Analytics

Supervisor: Dr. Jae Young Lee

BOSTON UNIVERSITY

Metropolitan College
Boston University

November, 2021

Declaration

This report has been prepared on the basis of our own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 4158

Student Name: Pascal Wyler

Contribution: Data Processing, Model Training and Evaluation

Student Signature: [illegible]

Student Name: Rayhan Aurelio

Contribution: Research and Exploratory Data Analysis

Student Signature:

Date of Submission: November 10th 2021

Table of Content

1. Abstract

Resource optimization in hospitals has been receiving a new found attention during the most recent pandemic. The COVID-19 pandemic has ignited a worldwide lack of resources among various industries, primarily medical-related ones. These constraints include but not limited to medical personnel, hospital beds, and various medical equipments. Given these limitations, medical institutions are under immense pressure to improve their patient triage capabilities. Therefore, considering the patient's chances of survival or the severity of their illnesses is indispensable.

This term project aims to predict hospital mortality from a sample data of 1177 patients who suffered from heart failure. Therefore, a binary label is estimated, indicating whether an individual may be more likely to survive or perish during the treatment process. The variables used for this classification task consist of demographic patient records, such as age, gender, and medical data collected from blood and urine samples.

In order to build an accurate classification model, this applied research project follows the standard data mining procedure. First, the patient records are examined during an initial data exploration phase which helps us identify any correlations between the features and target class. This supports the machine learning process, as these valuable attributes can largely explain the variance of the class attribute. Furthermore, it may prevent the model from over-fitting, given a large number of available attributes (47 to be exact). In order to overcome this issue, several feature selection methods are used to generate subsets of attributes that help the model to predict the hospital mortality accurately. The data subsets generated during this step are then used to train and test the following five classification models: Logistic Regression, Gaussian Naïve Bayes, Decision Tree, Random Forest, and Gradient Boost. Finally, the performance of these estimators is evaluated using a set of standard metrics such as precision, recall, and accuracy. In order to obtain rigorous results, the models are additionally trained using the "10-Fold cross-validation" method. From a technical perspective, the project is implemented using the Python library "sci-kit learn." The data pre-processing and feature selection steps are implemented using pipelines. Furthermore, it also contains a processing step of randomized over-sampling, given the imbalanced target attribute in the dataset.

The performance results of the 25 generated model instances indicates that classification accuracy of up to 89% can be achieved using random forest classification. However, the metrics also disclose that this model type has a particularly strong precision while showing a weak recall rate. Since mistaking a patient with low survival chances for a patient with good conditions can be understood as an ethical dilemma, due to this, the authors put more weight on recall. Consequently, the Gaussian Naïve Bayes classifier is found to be the best fit for the purpose, showing a recall rate of 78% and an overall accuracy of 81%. The selected features that contribute to this accuracy were 'Urine output', 'Leucocyte', 'Urea nitrogen', 'Blood calcium', 'Anion gap', and 'Bicarbonate'.

2. Data

The data used during this project originates from the public MIMIC-III Database, which contains medical records on heart failure patients that stayed at the Beth Israel Deaconess Medical Centre between the year 2001 and 2012 in the city of Boston, MA (Johnson, Alistair et al., 2015). The database extract has been used for previous research by a team from the Zhongshan Hospital based in Shanghai (Zhou, Jingmin et al., 2021). Their research data has been published on datadryad.org, from where they were retrieved for this project. The following paragraphs will introduce the class attribute and its most correlating features, and their distribution.

The class attribute "Outcome" tells us the condition of the patient, "0" for being alive and "1" for having been pronounced deceased. However, when looking at the distribution of the two classes, the target is found to be severely imbalanced with 1017 surviving patients and 159 deceased individuals. This issue will be addressed during data pre-processing by applying randomized over-sampling.

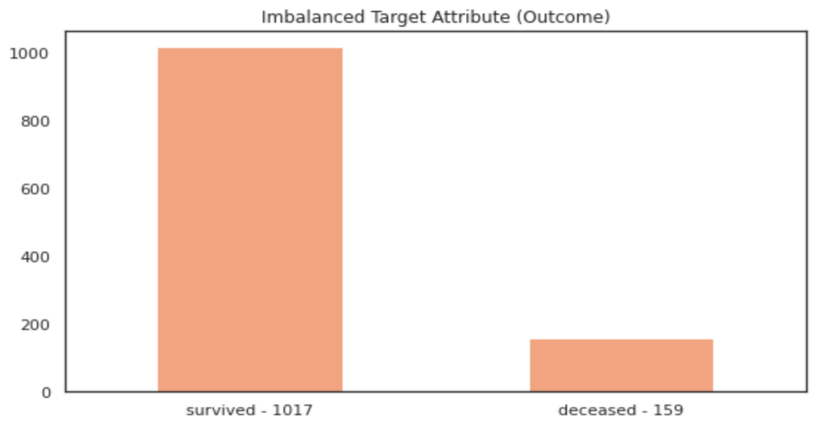


Figure 1: Imbalanced Target Class

When looking at the Pearson correlation between the attributes and the target class, nine variables show an above-average coefficient. Their interpretation is as follows. "Anion gap" with a value of "0.23" is the difference in the acidity level of the body; a higher anion gap is caused by the body producing a higher than normal amount of acid. "Leucocyte" with a value of "0.208" represents the number of white blood cells present in the body. "Urea Nitrogen" with a value of "0.203" is the amount of metabolic leftover in the body. Finally, "RDW" with a value of "0.148" is the red blood cell distribution width.

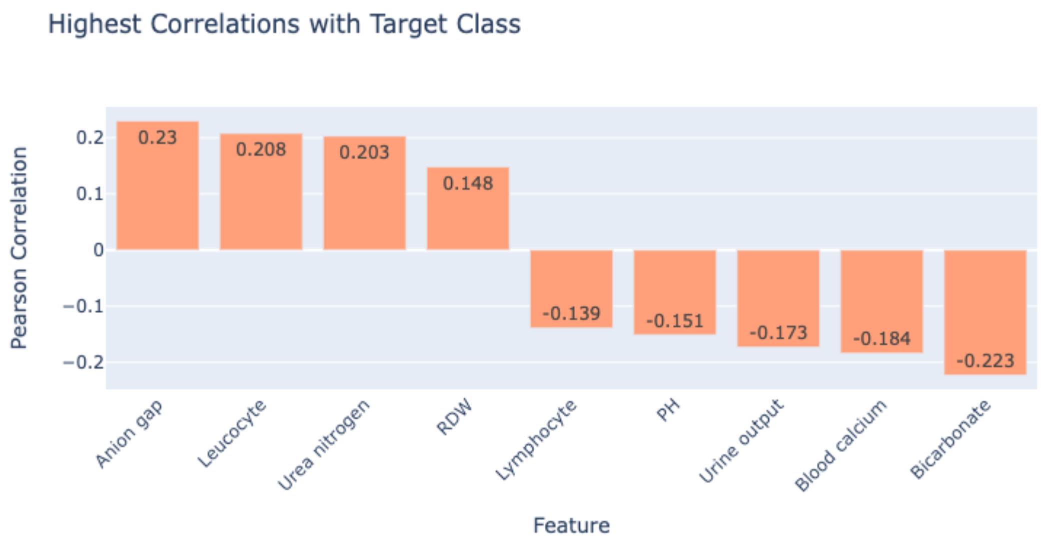


Figure 2: Features with highest Target Correlation

"Lymphocyte" with a value of "-0.139" represents the number of immune cells in the bone marrow, in this case meaning that there are simply lesser amounts of white blood cells than what is typically expected. "pH" with a value of "-0.151" represents the acidity/basicity levels of the body, in this case meaning that the body is simply more acidic, this is related to "lactic acid" which is one of the positively correlated features. "Urine output" with a value of "-0.173" represents the amount of urine produced by the body, in this case being negatively correlated, meaning that the amount of urine produced by the body is below what is normally expected. "Blood Calcium" with a value of "-0.184" represents the lack of Vitamin D in the body, medically known as "hypocalcaemia" which is usually caused by malnutrition (Lim & Clarke, 2014). Lastly, "Bicarbonate" with a value of "-0.223" represents the body's lower than expected amount of lactic acid. Further explanation regarding these aforementioned features along with the other features can be found in the appendix of this report.

In the dataset, "Anion gap" is a positively skewed feature, which means that there are roughly more people with a lower anion gap than average. "Leucocyte" is also a positively skewed feature, which means that there are more people with a lower white blood cell count in their body than average. "Urea Nitrogen" is also a positively skewed feature, which means that there are more people that have a lower amount of urea nitrogen than average. "RDW" is also a positively skewed feature, which means that there are more people that have a lower red blood cell distribution width than average. "Lymphocyte" is another positively skewed feature, which means that there are more people with a lower count of white blood cells than usual. "pH" is a symmetric feature, which means that the values for this attribute are evenly spread out. "Urine output" is another positively skewed feature which means that more people produce an abnormally small amount of urine than average. "Blood calcium" is symmetric, which means that the feature is evenly distributed. "Bicarbonate" is a slightly positively skewed feature, which means while the distribution is generally spread out evenly, there are more people with a less than average amount of bicarbonate which means that more people have a lower than expected amount of lactic acid.

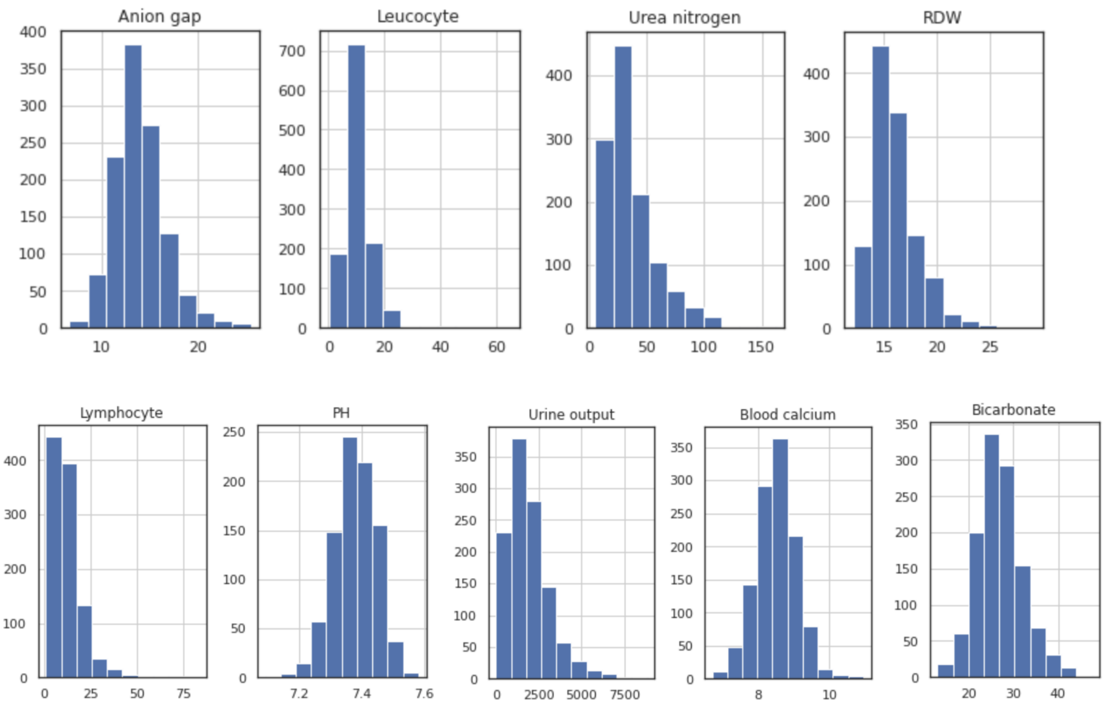


Figure 3: Feature Distributions

3. Data Mining Tools & Algorithms

The following paragraphs will describe the project implementation, tools, and algorithms used to predict hospital mortality. The data mining project is implemented using a Python 3 Notebook hosted on Google Colab. Moreover, it is organized to reflect the standard procedure of a data science project, which contains data exploration and pre-processing steps during the first stage. It is then followed by model selection, fitting, and prediction in a second step. In the final stage, the model outputs are evaluated using various performance metrics.

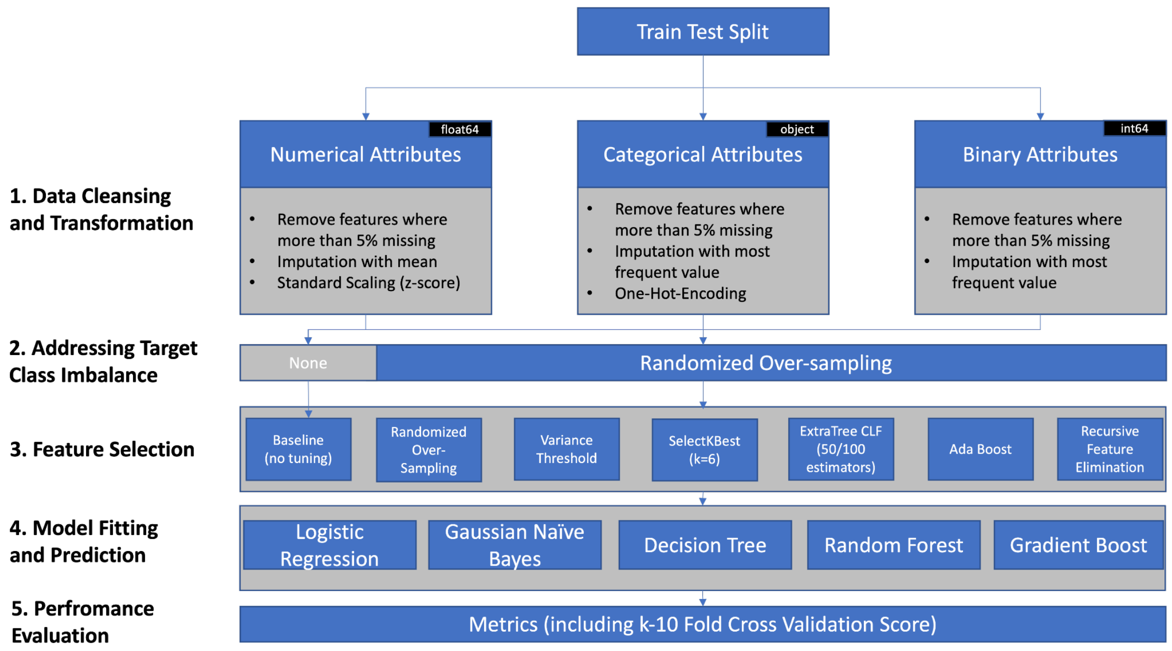
During the preliminary data exploration and wrangling phase, the libraries in use are Pandas, NumPy, Matplotlib, and Seaborn. While Pandas and NumPy are vital to importing and manipulating the dataset, Matplotlib and Seaborn is crucial in expressing the findings visually. The authors use sci-kit-learn and imbalanced-learn pipelines to maintain an organized code structure while building and evaluating models at scale. The benefit of implementing pipelines is that they can bundle multiple process steps in one function. More specifically, for this project, they combine data pre-processing with model fitting and tuning steps. Chapter 4 will provide more detail on the exact procedure.

To maximize the accuracy of the hospital-mortality prediction task, five different algorithms are implemented and evaluated using the sci-kit-learn library. Broadly they can be distinguished between single-classifiers and ensemble-learning models. The single-classifiers contain a Linear Regression model, a Gaussian Naïve Bayes classifier, and a decision tree. The ensemble-learning models consist of a Random Forest and a Gradient Boost classifier. It can be expected that the ensemble learning methods would come at a greater computational cost but would substantially outperform the single-learning algorithms in terms of accuracy and robustness. All the libraries used are summarized in the table below.

Library	**Description / Use Case**	**Reference**
Pandas; NumPy	Loading, filtering, and cleaning of the dataset	Pandas NumPy
Matplotlib; Seaborn	Visual exploration of the target class and attributes	Matplotlib Seaborn
Scikit-Learn	Bundling pre-processing, fitting, and tuning steps	scikit-learn
Imbalanced-Learn	Implementing strategies to address challenges of imbalanced datasets	imbalanced learn

4. Procedure

The following sections will discuss the data pre-processing, modelling, and tuning steps in detail. As outlined in Chapter 3, these steps are bundled as pipelines. The design is elaborated to train, test, and evaluate the 25 different model instances. These instances result from combining the five algorithms outlined in Chapter 3 and the five attribute selection methods that will be described in more detail in Chapter 5. Respectively, each model is implemented five times using a different attribute selection method during each iteration. The end-to-end process is illustrated below.



Train Test Split
float64
Numerical Attributes
Remove features where more than 5% missing
Imputation with mean
Standard Scaling (z-score)
object
Categorical Attributes
Remove features where more than 5% missing
Imputation with most frequent value
One-Hot-Encoding
int64
Binary Attributes
Remove features where more than 5% missing
Imputation with most frequent value
1. Data Cleansing and Transformation
2. Addressing Target Class Imbalance
None
Randomized Over-sampling
3. Feature Selection
Baseline (no tuning)
Randomized Over-Sampling
Variance Threshold
SelectKBest (k=6)
ExtraTree CLF (50/100 estimators)
Ada Boost
Recursive Feature Elimination
4. Model Fitting and Prediction
Logistic Regression
Gaussian Naïve Bayes
Decision Tree
Random Forest
Gradient Boost
5. Perfromance Evaluation
Metrics (including k-10 Fold Cross Validation Score)

Figure 4: Data Processing Pipeline

To pre-process the numerical, categorical, and binary data, three different transformations are applied to address the specialties of each attribute type in Step 1. First, missing numerical values are imputed with the median value of the feature. Then, after imputation, they are scaled using z-score standardization. Secondly, for missing categorical attributes, the imputation is done using the most frequent category. Finally, they are then transformed into dummy variables using the One-Hot Encoding method. Ultimately, the binary variables are imputed with the most frequent category. In contrast to numerical and categorical features, the binary attributes are not transformed. In general, any attribute where more than 5% of the data is missing is entirely excluded from the project as it could influence bias to the data. Consequently, the features ‘Lactic acid’ and ‘Basophils’ are dropped. Moreover, any outliers are removed using the 1.5 IQR Method. The transformation strategies for each feature type are then combined into one single column transformation instance, representing the first step of the pipeline.

```
# Preprocessing for numerical data
numerical_transformer = Pipeline(steps=[
    ('imputer',SimpleImputer(missing_values=np.nan,strategy='median')),
    ('StandardScaler',StandardScaler())
])


# Preprocessing for categorical data
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(missing_values=np.nan,strategy='most_frequent')),
    ('onehot', OneHotEncoder(sparse=False, handle_unknown='ignore'))
])

# Preprocessing for binary data
binary_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(missing_values=np.nan,strategy='most_frequent'))
])

# Bundle preprocessing steps
preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_cols),
        ('cat', categorical_transformer, categorical_cols),
        ('bin',binary_transformer, binary_cols)
    ])
```
Figure 5: Implementation of Data Transformation

In a second processing step, the randomized over-sampling strategy is added. As mentioned in Chapter 2, the target class shows a clear imbalance since the deceased patients are heavily underrepresented making up only 14% of all records. Therefore, this step is included in each pipeline.

As a third element, the pipeline contains the feature selection method as outlined in Chapter 5. To run and evaluate these methods separately, five different pipelines are built.

The fourth and last element of each pipeline is the model itself, passed to the "buildPipelines" function. A complete example can be found below. The same pipelines can be called and reused for each model as a result of this modular set-up, leading to the five-times-five instances. In order for the impact on the model's accuracy to be investigated easily, two additional pipelines are implemented. The first "baseline" pipeline only consists of the pre-processing steps and the model. It does not contain over-sampling and uses all features. The second additional pipeline is the "over-sampling" pipeline, which does not have any tuning but over-sampling.

```
my_pipeline2 = Pipeline(steps=[('preprocessor', preprocessor),
                              ('sampling', RandomOverSampler(random_state=100)),
                              ('feature_selection',SelectKBest(f_classif, k=6)),
                              ('model', model),
                             ])
```

Figure 6: Pipeline Implemenation in sci-kit learn

Once the models are passed to the pipelines, the new bundle can then be fitted and predicted using the “fitPredictPipeline” function in Step 4. It also directly calculates the accuracy score for a single-run as well as the means of the 10-fold cross-validation. Finally, using “getModelEvaluation” method, these performance metrics for each pipeline can then be printed as a heatmap (see Chapter 6). The functions are then called for each model type described in Chapter 3.

```
#fit-predict pipelines and return accuracy metrics
def fitPredictPipeline(my_pipeline):
  my_pipeline.fit(X_train, y_train)
  preds = my_pipeline.predict(X_valid)
  matrix = confusion_matrix(y_valid,preds)
  precision=precision_score(y_valid,preds)
  recall=recall_score(y_valid,preds)
  accuracy = accuracy_score(y_valid,preds)
  auc=roc_auc_score(y_valid,preds)
  cv_scores = cross_val_score(my_pipeline, x, y, cv=10,scoring='accuracy')
  mean_cv_score = cv_scores.mean()
  f1 = f1_score(y_valid,preds,average=None)[0]
  mcc=matthews_corrcoef(y_valid,preds)
  return precision,recall, accuracy,auc, mean_cv_score, f1,mcc, matrix,
   cv_scores, preds
```

Figure 7: Method to fit, predict and evaluate the classification models

5. Attribute Selection Methods

The following section will discuss how a subset of relevant features is being elaborated to improve on model performance. As outlined in Chapter 2, the dataset consists of 47 attributes in addition to one target class. To prevent over-fitting in the models, it will be reduced to a subset of features with maximum information gain in mind. The implementation is done by sci-kit-learn. The five different approaches to identifying this subset are as follows (1.13. Feature Selection, n.d.).

Variance Threshold: By defining a variance threshold, the target class distributions of each feature will be examined iteratively. The objective is to detect and eliminate features where there is only a low variance among its values. For instance, if a binary attribute with 100 values contains 95 "1"s and only 5 "0"s, it is less likely to help the model distinguish between target classes "0" and "1". In this case, the maximum variance threshold is set at 80%. Therefore, if a binary attribute consists of one class in more than 80%, it is removed from the sample set. However, the method was found only to consider categorical attributes. Hence, the features "CHD with no MI" ,"Depression" , and "COPD" are removed.

Select K Best with ANOVA F-Test: To remove attributes with little information gain, Select K Best searches for k optimal parameters to fit the model. In this project, k is randomly set as 6. Then, an ANOVA F-Test is conducted for each attribute to find the six best attributes to see whether there is a significant difference between target class "1" and "0". The features that have been selected are 'Urine output', 'Leucocyte', 'Urea nitrogen', 'Blood calcium', 'Anion gap', and 'Bicarbonate.

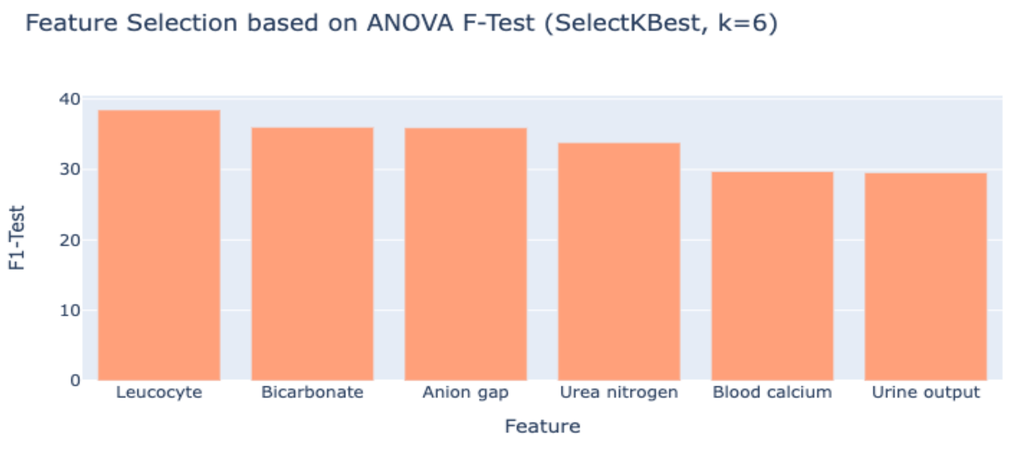


Figure 8: Selected Features by SelectKBest

Extra Tree Classifier: Before fitting the model, a decision tree is built to identify the impurity-based feature importance. The extra tree classifier then returns the highest nodes with the greatest information gain. This method is implemented twice, with a different number of estimators (n=50, n=100). The criterion used in both implementations is the Gini-Index. Respectively, the features with the highest Gini-Index are selected.

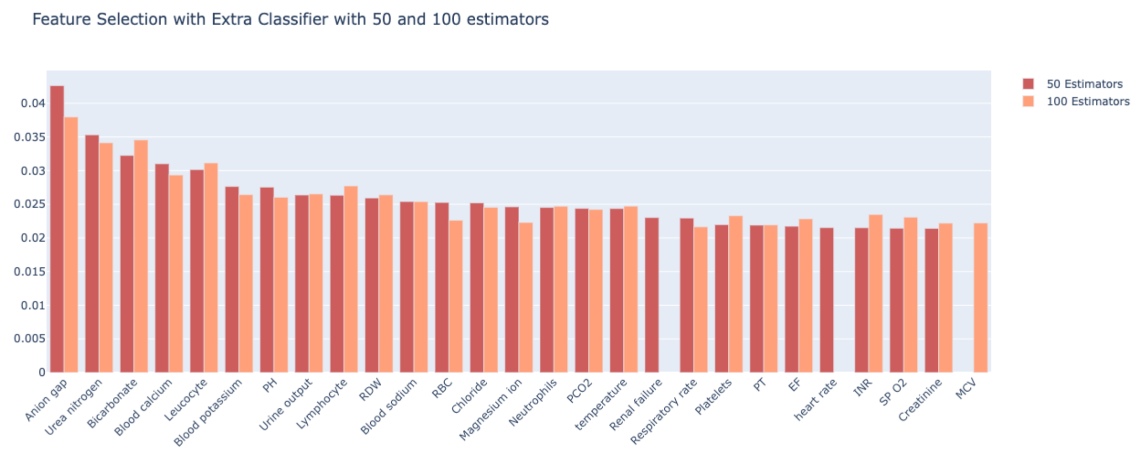


Figure 9: Selected Features by Extra Tree Classifier

AdaBoost: AdaBoost is an ensemble tree learning method that combines multiple "Weak Learners'" decisions. The decision power of each tree is weighted based on their split, respectively, their entropy values. Similarly, like with the Extra Tree Classifier, AdaBoost uses the Gini Index to measure the importance of each feature and selects them accordingly during feature selection. As part of the pipeline, the AdaBoost Classifier selects the 19 attributes with the highest index.

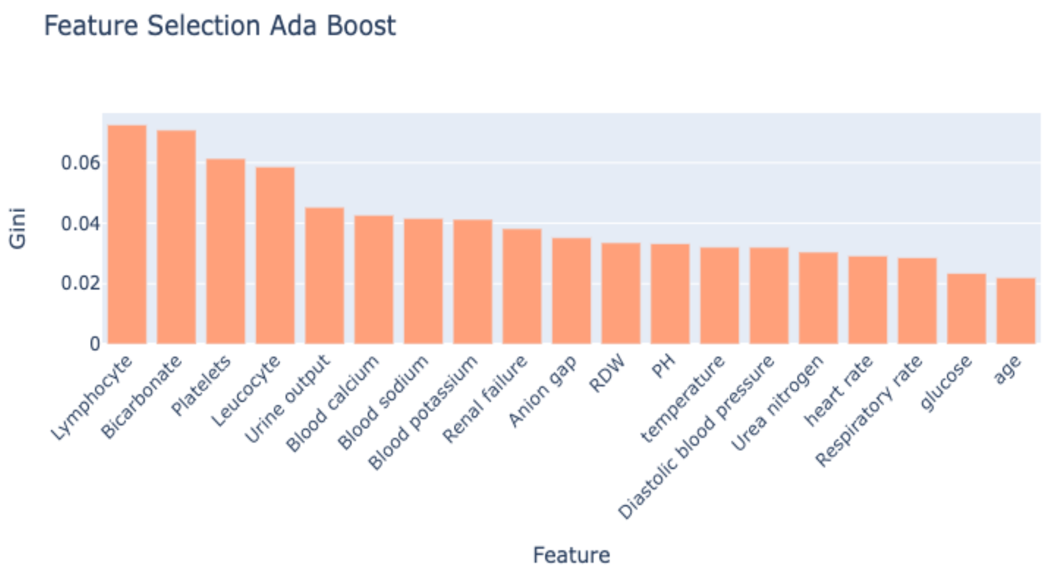


Figure 10: Selected Features by AdaBoost

Recursive Feature Elimination (RFE): When selecting the optimal subset of attributes using RFE, features are continuously removed from the training set. To assist with this process, an external estimator is successively trained and evaluated. Every time a feature is removed, the new accuracy is compared to the model performance before removing the particular attribute. If the removal increases the model performance, the feature is eliminated; otherwise it will remain in the subset. A logistic regression model is used to assist the process when selecting the attributes to predict hospital mortality. The number of selected features is set at k=10.

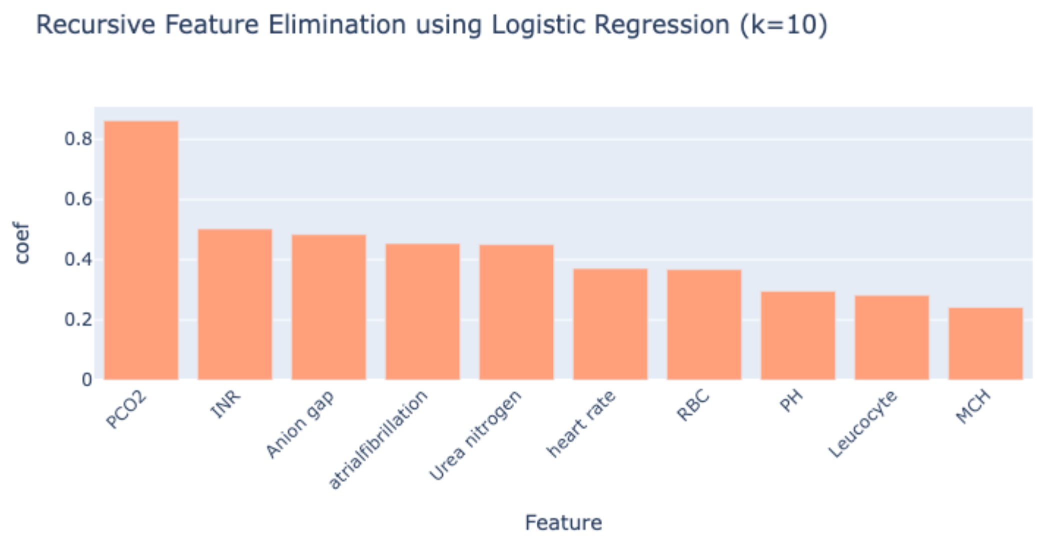


Figure 11: Selected Features by RFE

When comparing the outputs of different feature selection methods, it becomes evident that most of the approaches pick a similar subset of features. Especially, the attributes that were found indicates a comparably high correlation with the target attribute during the data exploration phase which appears frequently among the subsets. Examples include anion gap, urea nitrogen, leucocyte, bicarbonate, blood calcium, and urine output. These five different subsets will be used to train and evaluate the five classifier algorithms introduced in the previous chapter. The resulting 25 model instances will then be compared to identify which attribute selection method suits the classification task best.

6. Results and Evaluations

To get a holistic view of the performance metrics of the 25 model instances, the values are aggregated and visualized as heatmaps. A separate graph for each evaluation metric is used to display the best performing combination of model and feature selection. The metrics used to highlight the differences are as follows:

- Precision & Recall
- Accuracy & Area under the Curve (AUC)
- Mean value of 10-Fold Cross-Validation score
- F-1 Measure
- Matthews Correlation Coefficient (MCC)

Generally, since the F-1 and MCC scores are more practical when evaluating imbalanced datasets, their relevance becomes more important when looking at the baseline models in Column 1. However, all the other combinations in Columns 2 to 7 have an equal share of positives and negative outcomes thanks to the randomized over-sampling. Consequently, more attention will be dedicated towards Accuracy, AUC, Precision, and Recall.

When comparing the different heatmaps, a strong dominance of the random forest classifier can be observed. In particular, its combination with the attribute selection method "Extra Tree Classifier" with 50 estimators outperforms the alternative options when considering accuracy, F1-Score, and the Matthews Correlation Coefficient. Although when drilling deeper, it becomes evident that the random forest greatly relies on its precision rate, while its recall score is comparably low. The randomized over-sampling and feature selection methods appear to balance two metrics leading to an increased recall and lower precision score. Considering these aspects, the random forest classifier may be ranked as the number one model; when it comes to overall performance scores. Its fit-for-purpose, however, would only be detected if a patient survives compared to individuals with lower chances.

In the latter case, it would be more beneficial to inspect the performance metrics of the "Gaussian Naïve Bayes" classifier closely. Despite having a slightly lower accuracy rate than the random forest, its AUC and recall scores are higher than all other models. Consequently, it would be stronger at identifying high-risk patients while compromising its ability to spot a person with higher odds of survival. It generally performs best in combination with the feature selection methods "SelectKBest" (k=6) and the extra tree classifier with 50 estimators.

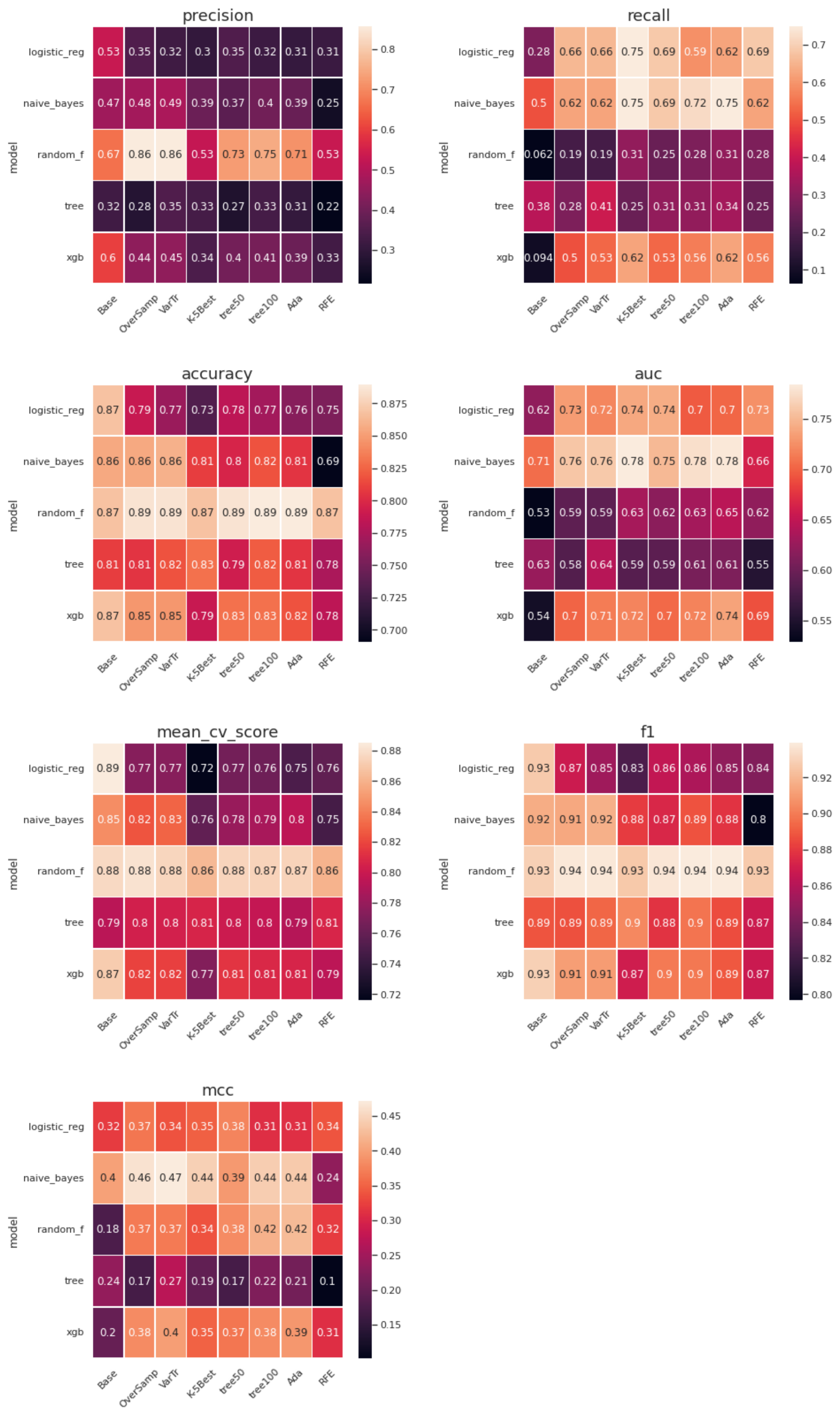


Figure 12: Evaluation Metrics

7. Discussion and Learnings

The use case of predicting hospital mortality among heart failure patients led to the discovery of relevant questions that must be addressed when prioritizing patients for treatment. It sheds light on an ethical dilemma that model engineers may face when balancing precision and recall rate. It also shows that a high accuracy, AUC, MCC or F-1 score alone does not answer a model's fit for purpose. Contrasting the performance metrics of Random Forest versus the ones of the Naïve Bayes classifier clearly indicates that that the latter option balances recall and precision better, therefore it is the better fit for predicting hospital mortality.

From an implementation standpoint, the project demonstrates how over-sampling can help to balance precision and recall and that a small set of strong features can outperform a large extensive collection of weak attributes. Comparing the baseline model instances with the over-sampled ones shows that the recall rate increases for four out of the five models. Furthermore, the attribute selection methods that pick fewer features like with "SelectKBest" (k=6) and "AdaBoost" (k=19) also largely contributed towards increasing the model's recall score.

To critically review the implemented models and findings, a number of limitations must be highlighted. First, the underlying dataset only provides a snapshot of the patient's health. Since the outcome of the medical samples may vary over time, feeding the model with longitudinal data may further improve on its performance. For further research, this experiment could be repeated with sample data taken at multiple points in time. A second limitation that is worth mentioning is the multicollinearity between the features. Given the scope and nature of this term project, the independence of the attribute has not been assessed. Lastly, the models would need to be deployed and tested in a live environment to evaluate their useability. By doing so, it would require finding out whether the model input data can be collected in a timely manner so that the prediction is available to the medical personnel when the prognosis is required. If this precondition is not given, the model could only be useful for post-mortem analysis, which would reduce its usefulness.

8. Appendix: Data Catalogue

Name	Description
ID	unique identifier for patient
outcome	survived = 0, deceased = 1; **Class attribute**
age	patient age
gendera	patient gender, 1=male, 2=female
BMI	Body Mass Index
hypertensive	high blood pressure
atrial fibrillation	irregular/rapid heart rhythm
CHD with no MI	Coronary heart disease (narrowing of small blood vessels) with no myocardial infarction (heart attack)
diabetes	patient has diabetes
Deficiency anemias	iron deficiency
depression	patient suffers from depression
Hyperlipemia	blood has too many lipids (or fats), such as cholesterol and triglycerides
Renal failure	kidney failure
COPD	Chronic obstructive pulmonary disease (respiratory disease)
heart rate	patients heart rate
Systolic blood pressure	maximum heart pressure during heart beat (top)
Diastolic blood pressure	pressure in arteries between the beats (lower)
Respiratory rate	breath per minute
temperature	body temperature
SP O2	blood oxygen level
Urine output	urometer
haematocrit	pct of red blood cells
RBC	red blood cells
MCH	mean corpuscular hemoglobin (protein in red blood cells that carries oxygen)
MCHC	mean corpuscular hemoglobin concentration (concentration in single blood cell)
MCV	Mean Corpuscular Volume (avg size of red blood cells)
RDW	Red cell distribution width
Leucocyte	white blood cells
Platelets	colorless cell fragments in blood that form clots and stop or prevent bleeding
Neutrophils	help fight infection by ingesting microorganisms and releasing enzymes that kill the microorganisms
Basophils	type of white blood cell
Lymphocyte	immune cell in bone marrow
PT	Prothrombin time test
INR	monitor individuals who are being treated with the blood-thinning medication
NT-proBNP	high value indicates heart failure A normal level of NT-proBNP, based on Cleveland Clinic's Reference Range is: Less than 125 pg/mL for patients aged 0-74 years. Less than 450 pg/mL for patients aged 75-99 year
Creatine	high creatinine levels indicate kidney issues
kinase	increased levels may indicate a heart attack
Creatinine	high creatinine levels indicate kidney issues
Urea	high levels indicate kidney injury or disease
nitrogen	high levels indicate kidney injury or disease
glucose	higher levels may indicate diabetes
Blood potassium	nutrient concentration in blood -> high levels indicate kidney injury or disease
Blood sodium	high levels may indicate Diarrhea, A disorder of the adrenal glands. A kidney disorder.
Blood calcium	high levels indicate nervous system problems
Chloride	high levels indicate dehydration
Anion gap	acid produced by the body
Magnesium ion	magnesium concentration
PH	acidity or basicity values
Bicarbonate	body is having trouble maintaining its acid-base balance
Lactic acid	increase with low oxygen levels
PCO2	The partial pressure of carbon dioxide (PCO2) is the measure of carbon dioxide within arterial or venous blood
EF	low values may indicate heart failure

9. Bibliography

1.13. Feature selection. (n.d.). Scikit-Learn. Retrieved November 6, 2021, from https://scikit-learn/stable/modules/feature_selection.html

Johnson, Alistair, Pollard, Tom, & Mark, Roger. (2015). *MIMIC-III Clinical Database* (1.4) [Data set]. PhysioNet. https://doi.org/10.13026/C2XW26

Lim, V., & Clarke, B. L. (2014). Hypocalcemia. In F. Bandeira, H. Gharib, A. Golbert, L. Griz, & M. Faria (Eds.), *Endocrinology and Diabetes: A Problem-Oriented Approach* (pp. 265–278). Springer. https://doi.org/10.1007/978-1-4614-8684-8_21

Zhou, Jingmin, Li, Fuhai, Song, Yu, Fu, Mingqiang, Han, Xueting, & Ge, Junbo. (2021). *Prediction model of in-hospital mortality in intensive care unit patients with heart failure: Machine learning-based, retrospective analysis of the MIMIC-III database*. https://doi.org/10.5281/ZENODO.5032847

10. Table of Figures