We are pleased to have the opportunity to present our findings from the analysis of the heart failure data. Our work involved using advanced data science techniques to predict a patient's survival outcome and to provide insights to medical professionals about the medical features that impact heart failure. This work was done in two phases. In the first phase we pre-processed the original data given and in the second phase, we utilized the pre-processed dataset with selected features to train and build models aimed at making accurate predictions.

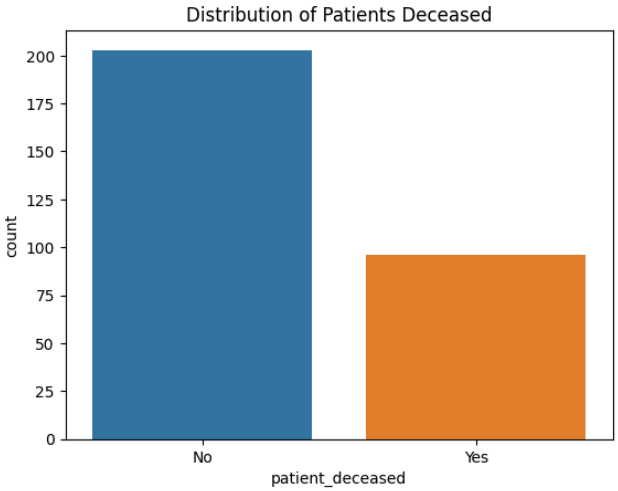
When we began to analyze the data from the 299 patient records that were present in our heart failure dataset, we first cleaned our dataset by removing rows with duplicate entries of patient records, rows containing missing values and outliers or anomalies in the dataset. This step is pivotal because it allows us to avoid obtaining misleading statistical results. In addition, it improves the accuracy of the predictions made determining if a heart failure patient will survive or not based on their medical information. As observed in Table 1, the original dataset consists of 13 columns of medical information of a typical heart failure patient. The first 12 features are attributes or individual pieces of information that represent each patient, including factors such as age, creatine phosphokinase levels, ejection fraction, and smoking status. Feature 13, 'patient deceased', is the target attribute to be predicted and is represented by two classes with 0 signifying that the patient survived and 1 representing the patient's death.

Table 1 Dataset Information

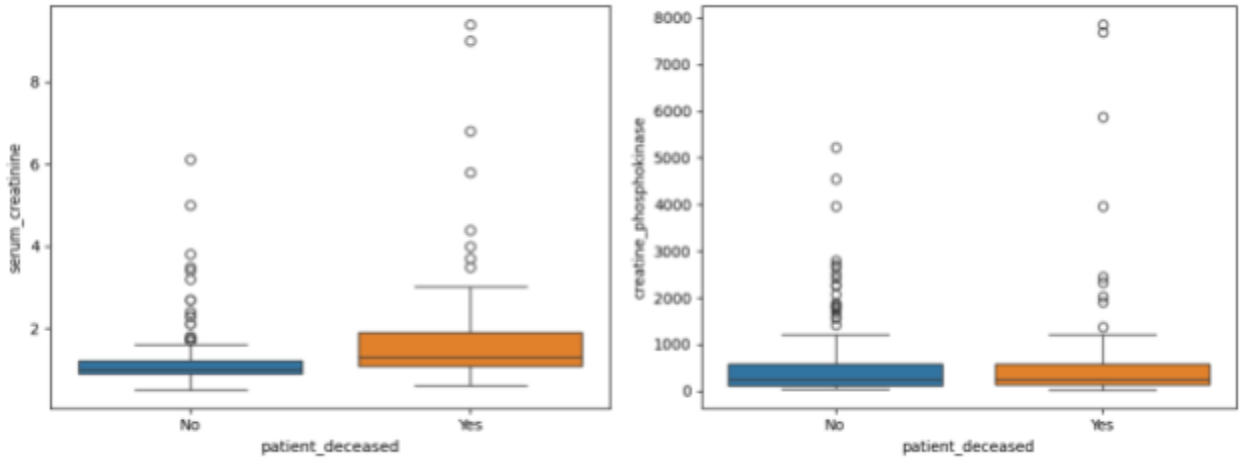
#	Feature	Description	Data Type	Domain
1	Age	Patient's age	Integer	[40, 95]
2	Anemia	Presence of anemia	Boolean	0, 1
3	Creatine Phosphokinase	Level of enzyme in blood	Integer	[23, 7861]
4	Diabetes	Presence of diabetes	Boolean	0, 1
5	Ejection Fraction	Percentage of blood ejected per heartbeat	Integer	[14, 80]
6	High Blood Pressure	Presence of high blood preassure	Boolean	0, 1
7	Platelets	Platelet count in blood	Float	[25100, 850000]
8	Serum Sodium	Level of sodium in blood	Integer	[113, 148]
9	Serum Creatinine	Level creatinine in blood	Float	[0.5. 9.4]
10	Sex	Patient's gender	Boolean	0, 1
11	Smoker	Smoking status	Boolean	0, 1
12	Days Until Follow-Up	Follow-up period	Integer	[4. 285]
13	Patient Deceased	Patient died	Boolean	0, 1

Data analysis was further performed in order to understand the underlying distribution of each patient grouped based on the outcome of whether the patient survived or not. This is important as it allows us to determine if there are any presence

of outliers or unexpected anomalies present in the data that we need to remove. In order to perform this visualization techniques such as box plots, bar charts and heat maps. Based on our observations there is a significant imbalance between the number of patients in each target class. This is significant in determining how we choose the train our machine learning models explained later.

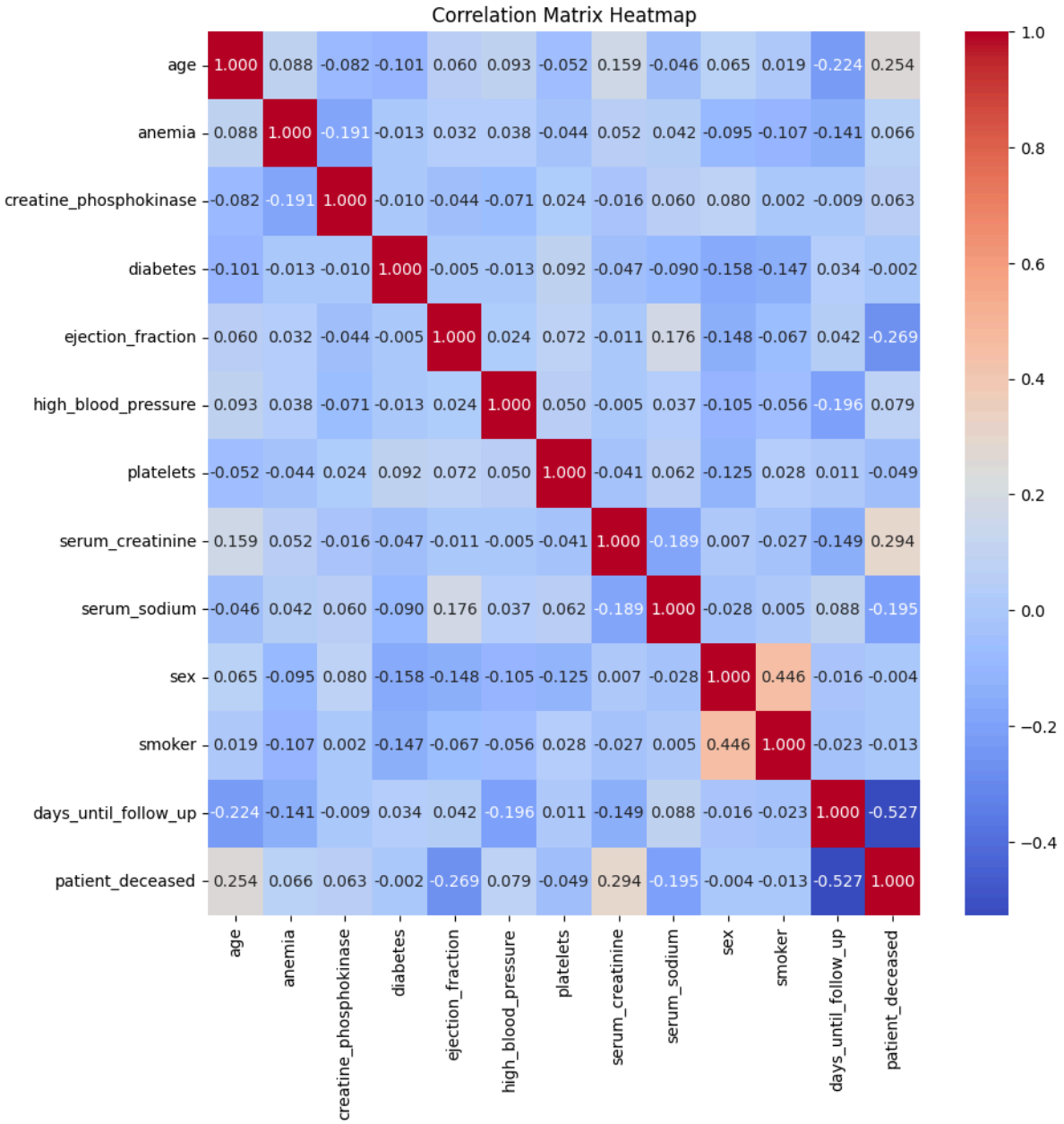


Box plots were primarily used to detect the presence of anomalies in our dataset that may have a detrimental effect on the accuracies of our models. Outliers present can possibly be present due to external factors such as abnormal conditions, measurement errors or incorrect data entry. As shown in the figure, serum creatinine and creatine phosphokinase box plots are provided with the circles present representing outliers. It is important to acknowledge that these outliers might hold significance in training accurate models. To determine their significance we sought out research articles to observe typical ranges medical professionals tend to use for these medical feature columns.



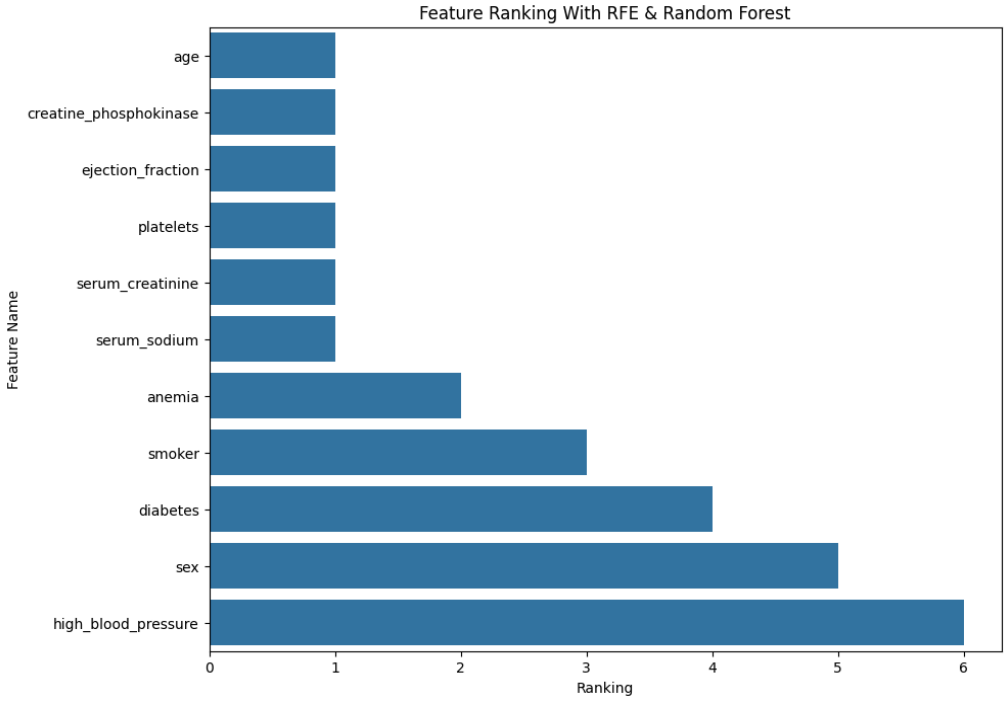
After research was performed we were able to come to the conclusion that the outliers could be safely removed. The outlier detection technique used is determining the lower and upper bound using the formula Q1 - 1.5 * IQR and Q3 + 1.5 * IQR respectively. This

technique was utilized because the box plots are not symmetric and skewed. For skewed distributions, the mean and standard deviation of each attribute are heavily influenced by outliers.

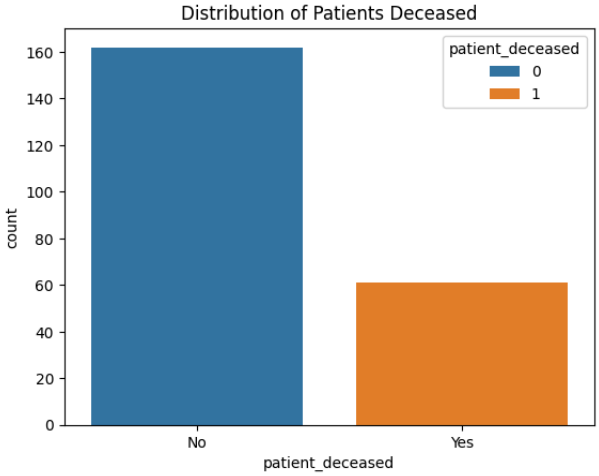


Heat maps were used to observe if there were underlying relationships between each column in the dataset. Ideally all columns should be independent from one another. Allowing relationships to be present leads to redundancy and the one of the related columns can be removed because they represent the same outcome. As we can observe from the heat map, all columns in our dataset are independent of one another.

However, we were able to determine that not all columns are relevant when predicting the survival of the heart failure patient through the use of machine learning algorithms. An algorithm called Recursive Feature Elimination was used to rank which features or columns are most relevant in predicting the patient's survival. It is utilized by training a machine learning algorithm, in our case, Random Forest Classifier, to obtain the importance of our features. This process works by removing the features with the least importance, and then repeatedly re-trains the model with the remaining



features until the desired number of features remains. As observed in the figure, age, creatine phosphokinase, ejection fraction, platelets, serum creatinine, serum sodium and anemia are the most relevant columns. By removing the irrelevant columns and abnormalities in the original dataset we were able to create a new dataset that consisted of 224 rows of patient medical records and 8 feature columns. This is significant



because it reduces the training time of our machine learning models.

After preprocessing was finished, our next step was to develop a machine learning model that will assist medical professionals to accurately predict a heart failure patient's survival. To perform this, classification machine learning algorithms are used to assign a class of label of 0 or 1 given previously unseen heart failure patient medical data. As mentioned previously a class label of 0 signifies the patient survives and 1 signifying the patient's death. These algorithms are able to learn from the data and gradually improve its performance by improving from its prior mistakes. After extensive research, we chose 4 machine learning algorithms and compared the results of each model based on different scoring metrics to determine which one was best to use for our data.

These models consisted of Logistic Regression, K-Nearest Neighbors, Random Forests and Gradient Boosted Trees. Each algorithm has its own unique strengths and weaknesses that we had to account for in order to determine the best model for our task. Initially the dataset we generated from the original was divided into training and testing sets. This step is pivotal because in order to evaluate the overall effectiveness of our models, we need to observe how well each trained model performs on unseen data and in our case the testing set. First each machine learning model was trained by adjusting its hyperparameters or configurations because performance is highly dependent based on which initial configurations are chosen. After determining which hyperparameter values were best for each model, we trained the models on the training set. The code presented provides a strict overview of how we were able to split the preprocessed dataset, train, configure and evaluate our models using the Python library, Scikit-learn.

```
df = pd.read_csv('data/preprocessed_heart_failure_dataset.csv')

X = df.copy().drop(columns=['patient_deceased'])
y = df['patient_deceased'].copy()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,
random_state=42)
```

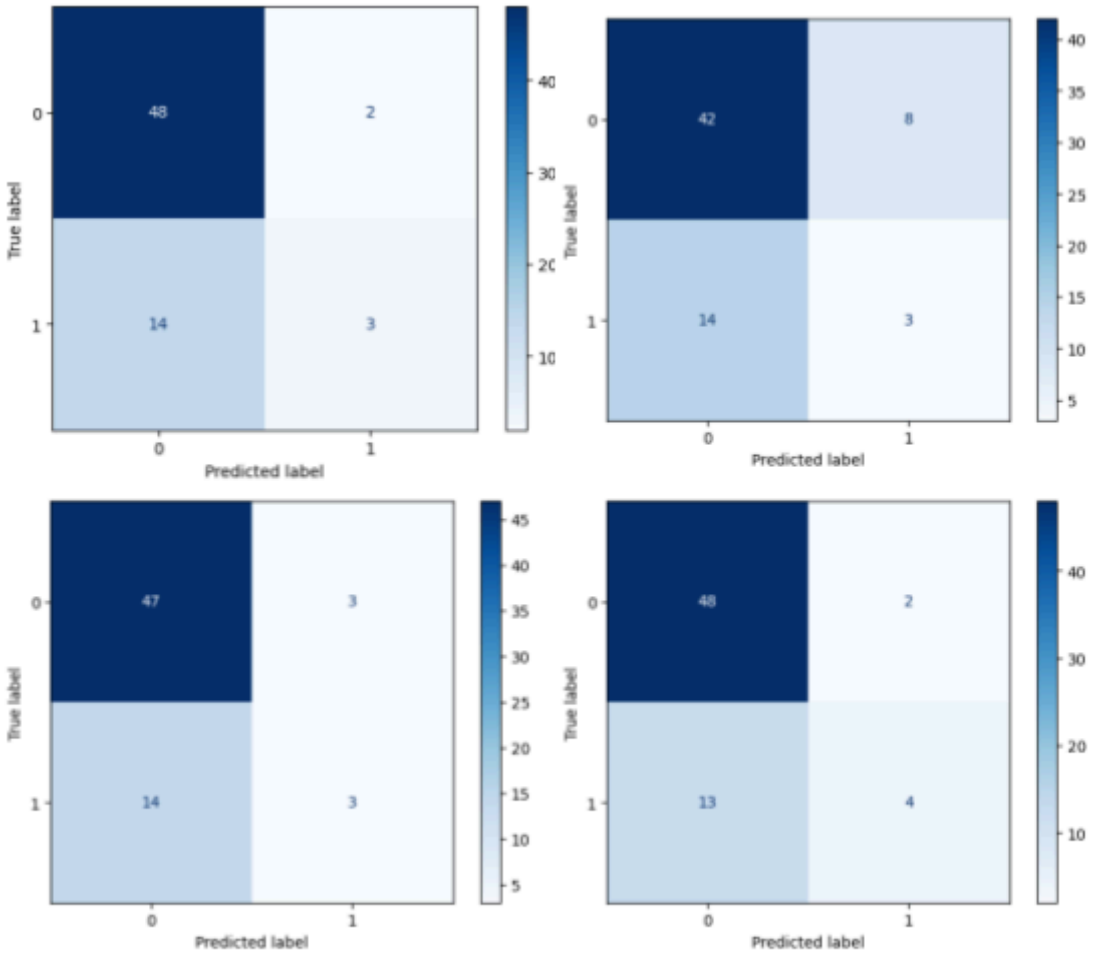
```
gbc_pipeline = Pipeline([
    ('scaler', standardizer),
    ('gbc', GradientBoostingClassifier())
```

```
])

param_grid = {
    'gbc__n_estimators': [100, 200, 300],
    'gbc__learning_rate': [0.01, 0.1, 1.0],
    'gbc__max_depth': [3, 4, 5],
    'gbc__min_samples_split': [2, 3, 4],
    'gbc__min_samples_leaf': [1, 2, 3],
    'gbc__subsample': [0.5, 0.75, 1.0],
    'gbc__max_features': ['sqrt', 'log2', None]
}

stratkf = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)

scoring = {
        'accuracy': make_scorer(accuracy_score),
        'recall': make_scorer(recall_score),
        'precision': make_scorer(precision_score, zero_division=1),
        'f1': make_scorer(f1_score),
        'roc_auc': make_scorer(roc_auc_score, needs_proba=True)
}

gbc_rs_search = RandomizedSearchCV(gbc_pipeline,
param_distributions=param_grid, scoring=scoring, refit='accuracy',
cv=stratkf, n_jobs=-1)
gbc_rs_search.fit(X_train, y_train)
```

After training of the machine learning models are complete, it is now time to test out the performance of these models using a confusion matrix using the test set consisting of previously unseen data. This is important as machine learning models need to be generalized, having strong metrics only on the training set is insufficient in determining how well a model predicts unseen data. A confusion matrix displays the number of true positives, true negatives, false positives, and false negatives of our machine learning model predictions. In our case a true positive is correctly predicting a patient's survival and a true negative is correctly predicting the patient's death. A false positive is

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

incorrectly predicting a patient's survival, but in reality the patient is deceased. Similarly, a false negative is incorrectly predicting the patient's death, but in fact the patient is still alive. The confusion matrix provides metrics of accuracy, precision, recall, F1 score and receiver operating characteristic (ROC) area under the curve (AUC) to evaluate which model performed the best. The best machine learning model chosen for our use case will have the highest accuracy and ROC AUC. Accuracy represents the proportion of correct predictions out of all the predictions made. ROC AUC compares the rate of true positives vs false positives, low values of this metric may result in



Top Left: Linear Regression, Top Right: KNN, Bottom Left: Random Forests, Bottom Right: Gradient Boosted Trees

unwanted situations where patients are wrongly classified to succumb to death.
We can observe from the test metrics, the Gradient Boosted Trees algorithm provides the best results in term of accurately predicting the survival or death of a patient and

	Accuracy	Precision	Recall	F1	ROC AUC	Model
0	0.761	0.600	0.176	0.273	0.568	Logistic Regression
1	0.672	0.273	0.176	0.214	0.508	KNN
2	0.746	0.500	0.176	0.261	0.558	Random Forest Classifier
3	0.776	0.667	0.235	0.348	0.598	Gradient Boosted Tree

preventing misclassifications of death. As a result in terms of both model complexity and test metrics, the Gradient Boosted Tree model is best suited for our task. The advantage of using the Gradient Boosted Tree model is that it can capture complex relationships between the features in our dataset, while still having strong performance. Gradient Boosted Trees is able to overcome the weaknesses such as high variability on unseen data by producing an overall strong model composed of multiple weak models with poor accuracies.

The significance of our results lies in improving heart failure patient care by assisting medical professionals to accurately predict a heart failure's patient survival based on their current medical records. With the data obtained from statistical analysis, health professionals are able to accurately determine which metrics are most prevalent in causing patient death. This can help medical professionals develop treatment plans and have a general overview of what important metrics to monitor during patient checkup. Our current model created with Gradient Boosted Trees is trained with a very small dataset of around 200 entries, but possesses an accuracy of 77.6 %. With larger amounts of data available, we can develop a more accurate model by further configuring the Gradient Boosted Tree model. An advantage of using the model we created is that it can be used in periodic polling where a patient's records are currently monitored and preplanned actions can be developed before a patient's vitals reach a certain threshold causing death. Health professionals could also use this model to provide personalized care for high risk patients by tailoring treatment plans, medication regimes, and follow-up schedules based on the patient's degree of risk.