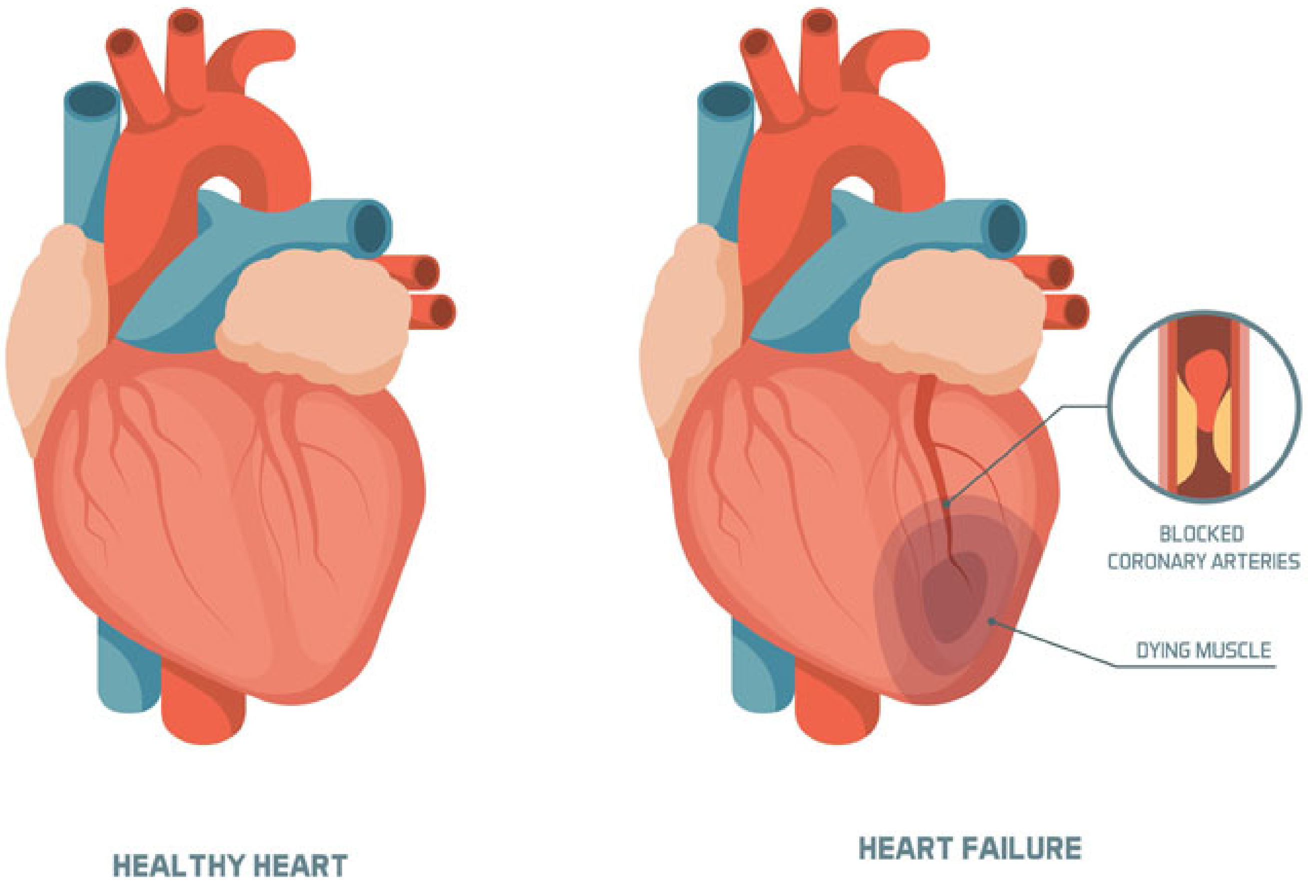
Heart Failure Prediction

Rishabh Jain, Saurav Kumar, Sarthak Ratnakar Deore
B20EE083, B20EE081, B20EE058

Abstract: This paper reports our experience with building a Heart Failure Predictor. We have a dataset containing certain features for patients and whether they have a heart disease. The dataset has failure represented by 1s and healthy heart represented by 0s.

I. INTRODUCTION

Heart failure is a common event caused by Cardiovascular diseases (CVDs) and this dataset contains 11 features that can be used to predict possible heart disease. People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidemia, or already established disease) need early detection and management wherein a machine learning models can be of great help.



Datasets:
heart: The file heart.csv is used as the dataset.
The train dataset contains 918 rows where each row represents a medical conditions of a patient in columns containing :

- Age
- Sex
- ChestPainType
- RestingBP
- Cholesterol
- FastingBS
- RestingECG
- MaxHR
- ExerciseAngina
- Oldpeak
- ST_Slope
- HeartDisease

The dataset has been split into train and test with test size of 0.3

II. METHODOLOGY

OVERVIEW
There are various classification algorithms present out of which we shall implement the following:

- DecisionTreeClassifier
- MLPClassifier
- XGBClassifier
- LGBMClassifier
- RandomForestClassifier
- AdaBoostClassifier
- KNeighborsClassifier
- SVC
- GaussianNB
- LogisticRegression

Exploring the dataset and pre-processing:

On counting the number of NULL values in the train dataset , it was found that there are no NULL values present.
The dataset's features were then standardized using a custom built function before training.
The data was then split into train and test sets with test size of 0.3.
The categorical features' 'Sex','ChestPainType','RestingECG','ExerciseAngina','ST_Slope' into Ordinal Encoding using OrdinalEncoder preprocessing technique.

Implementation of classification algorithms:

Each module was imported and fitted into a ML pipeline.

- *Desicion Tree Classifier:* The decision tree classifier creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.

- *MLPClassifier:* MLPClassifier trains iteratively since at each time step the partial derivatives of the loss function with respect to the model parameters are computed to update the parameters. It can also have a regularization term added to the loss function that shrinks model parameters to prevent overfitting. This implementation works with data represented as dense numpy arrays or sparse scipy arrays of floating point values.

- *XGBClassifier:* XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.

- *LGBMClassifier:* LightGBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages:
    1. Faster training speed and higher efficiency.
    2. Lower memory usage.
    3. Better accuracy.
    4. Support of parallel and GPU learning.
    5. Capable of handling large-scale data.

- *RandomForestClassifier:* Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

- *AdaBoostClassifier:* An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

- *KNeighborsClassifier:* The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It’s easy to implement and understand, but has a major drawback of becoming significantly slows as the size of that data in use grows. But, since our dataset size is small, it can be used comfortably.

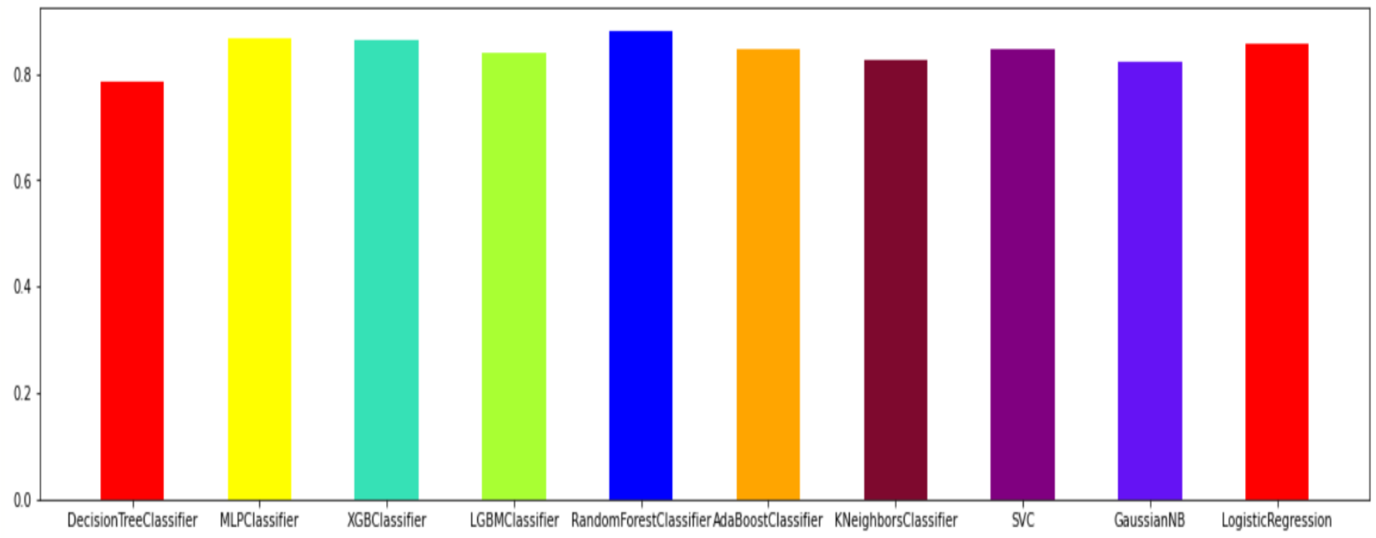
- SVM: The most applicable machine learning algorithm for our problem is Linear SVC. The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data.

- *GaussianNB:* The Gaussian Processes Classifier is a classification machine learning algorithm. Gaussian Processes are a generalization of the Gaussian probability distribution and can be used as the basis for sophisticated non-parametric machine learning algorithms for classification and regression

- *LogisticRegression:* Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

III. CLASSIFICATION ON TEST SET

The bar plot of the Accuracy score v/s Model is shown below:



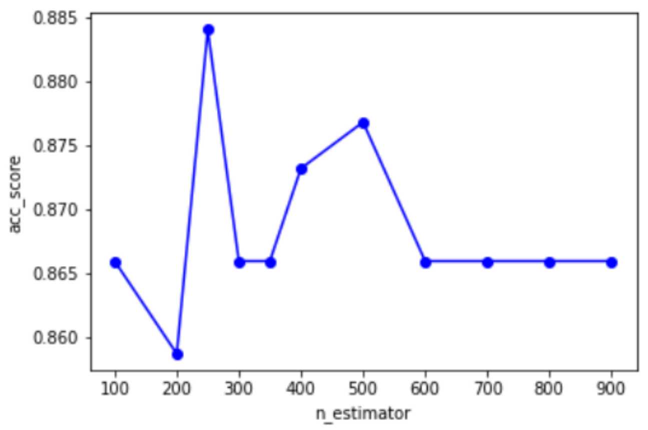
IV. PARAMETER HYPERTUNING

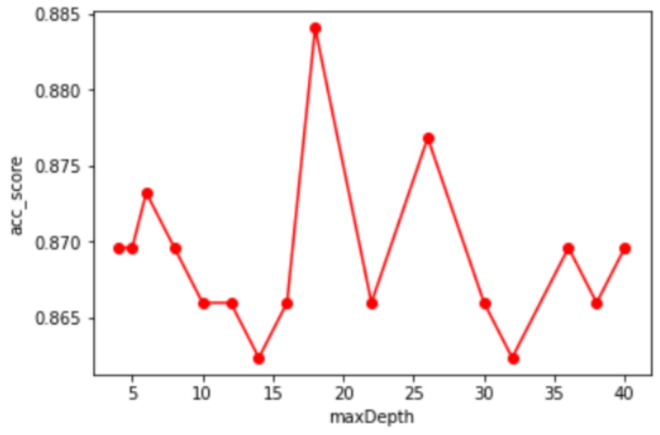
We get the best accuracy for the given following models so we will hyper tune the following 5 models using GridSearchCV:

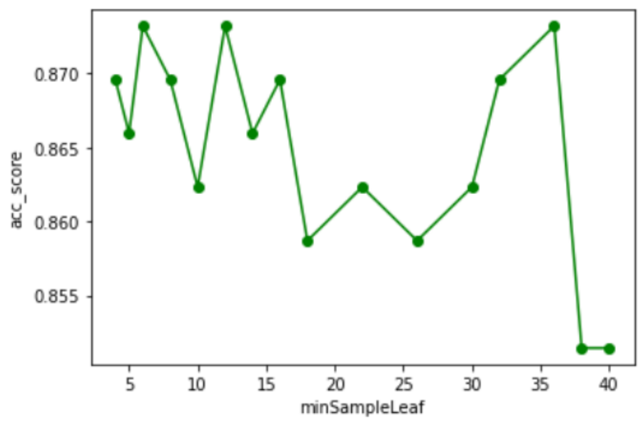
1) RandomForestClassifier
2) MLP
3) XGBClassifier
4) LogisticRegression
5) SVM

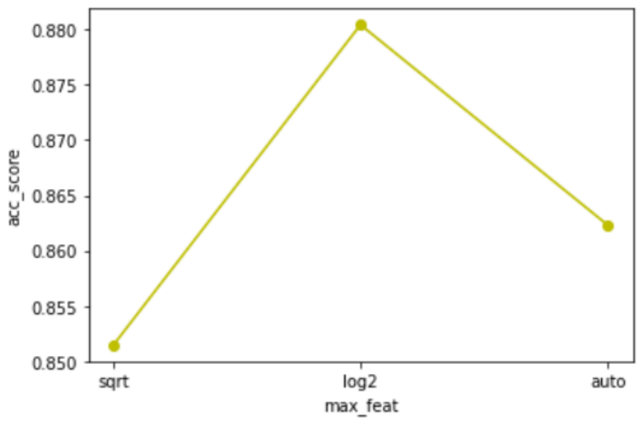
V. EVALUATION OF MODELS

1. RandomForestClassifier: The parameters n_estimators, maxDepth, minSampleLeaf and max_feat were varied and accuracy scores were plotted.

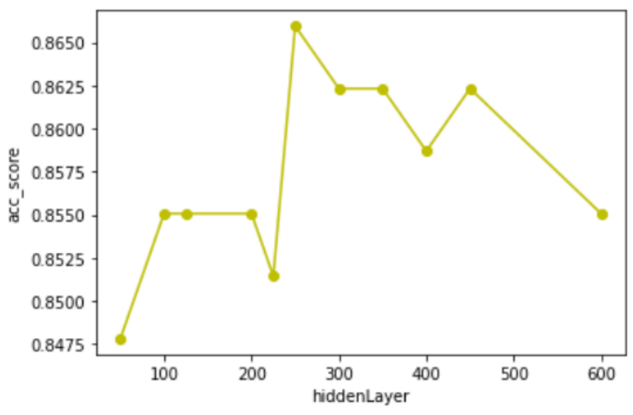


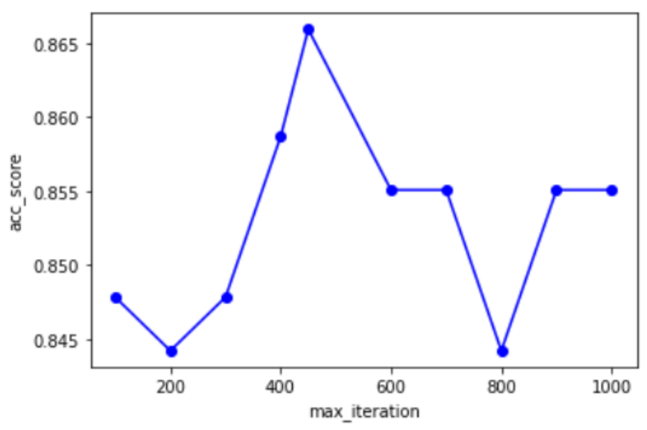


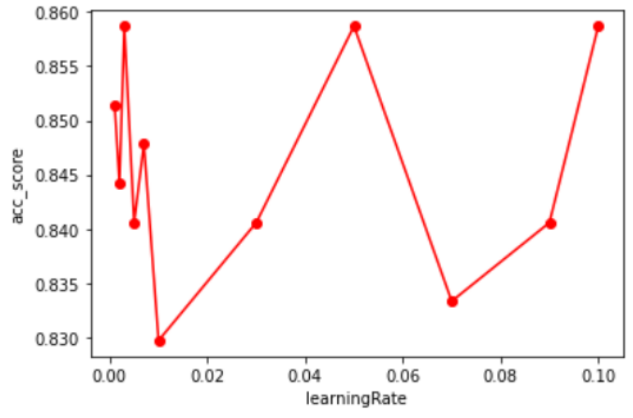




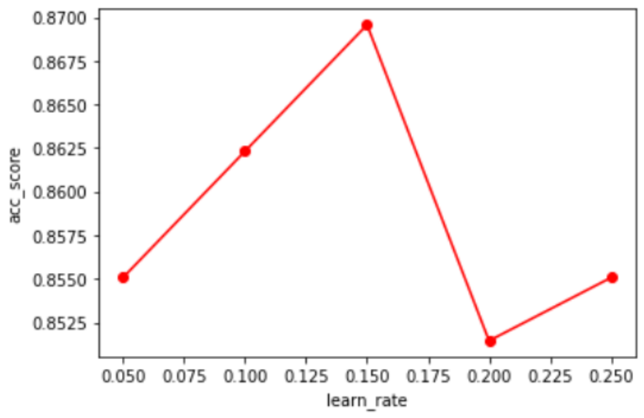
2. MLP: The parameters hiddenLayer, max_iteration and learningRate were varied and accuracy scores were plotted.

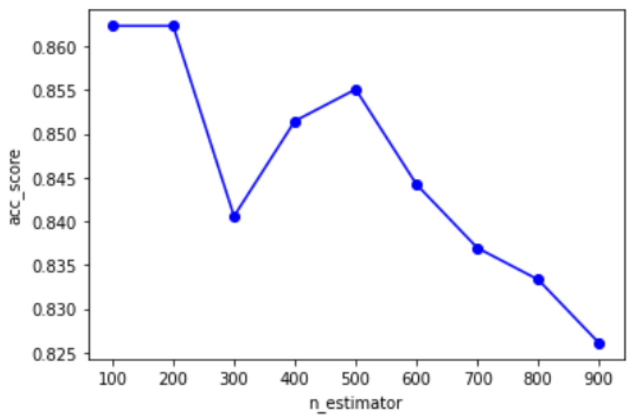


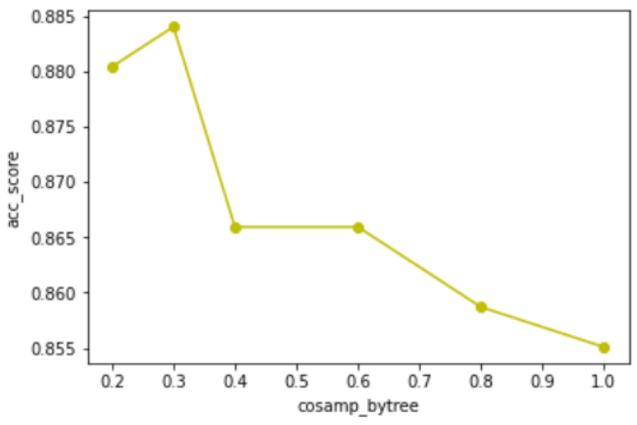




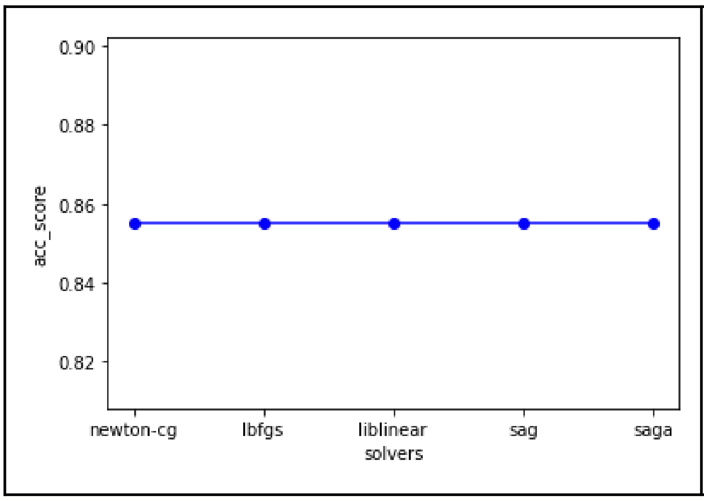
3. XGBClassifier: The parameters learn_rate, n_estimator and cosamp_bytree were varied and accuracy scores were plotted.

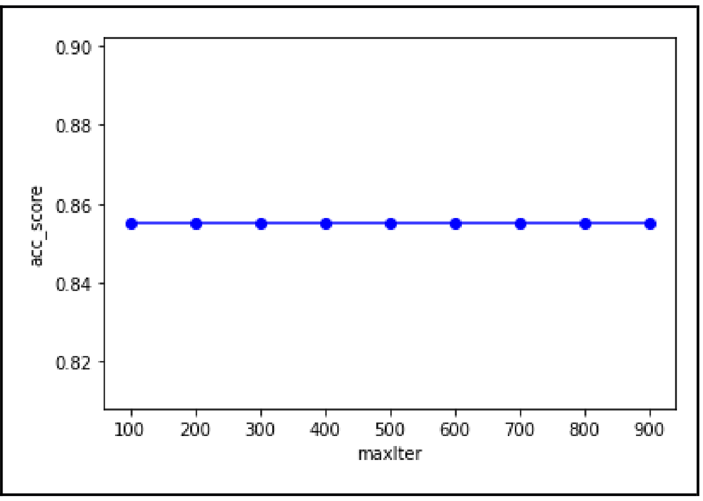




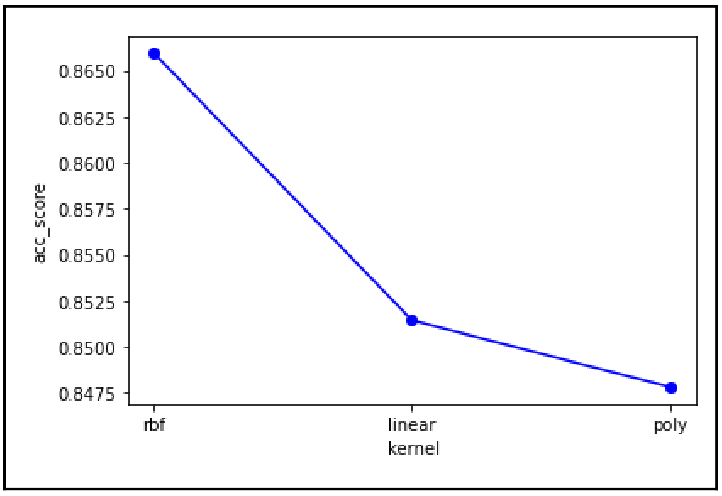


4. LogisticRegression: The parameters solvers and maxiter were varied and accuracy scores were plotted.





5. SVM: The parameter kernel was varied accuracy scores were plotted.



VI. RESULTS AND ANALYSIS

The accuracy score plots shows that all classifiers had nearly equally efficient performance. Models like RandomForest and DecisionTree took lesser computational time. However, other models like SVM are strong enough and their speed was hardly affected.

The technique of hyper parameter adjustment (optimization) is an important part of machine learning. A proper selection of hyperparameters might help a model achieve the intended metric value or, on the other hand, can lead to an endless cycle of training and optimization.

CONTRIBUTIONS:

The learning and planning was done as a team.The individual contributions are as given

- Saurav Kumar (B20EE081): Background research and training the models
- Rishabh Jain (B20EE083): HyperTuning and synthesizing the results
- Sarthak Ranakar Deore (B20EE058): Preprocessing data and report making

REFERENCES:

scikit-learn.org

PRML lecture notes (Dr. Richa Singh)

Kaggle dataset (https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction)