Heart Failure Prediction Sudip Kumar (B20CS072)

Data set:

https://drive.google.com/file/d/1osorOQ4uUz-iqpAJ1rwfiJY4xzcyU9e9/view?usp=sharing

I have done the project by going through these several steps:

* Step 1: Import the provided data set in the colaboratory file by importing pandas.
* Step 2: Go through the preprocessing procedure.
* Step 3: Perform data visualization
* Step 4: then plotted the graph for keeping X- axis as age and Y-axis as the

disease by importing matplotlib.pyplot

* Step 5: I have done the train and test splitting by importing train test split
* Step 6: Started with implementing different model applications. The models

are as follows:

* 1. from sklearn.linear\_model import LogisticRegression
     + In the multiclass case, the training algorithm uses the

one-vs-rest (OvR) scheme if the ‘multi\_class’ option is set to ‘ovr’, and uses the cross-entropy loss if the ‘multi\_class’ option

is set to ‘multinomial’.

* This class implements regularized logistic regression using the ‘liblinear’ library, ‘newton-cg’, ‘sag’, ‘saga’ and ‘lbfgs’ solvers. **Note that regularization is applied by default**.
* the accuracy using this model is 0.855072463768116
  1. from sklearn.ensemble import BaggingClassifier
     + A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions to form a final

prediction.

* + - Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator, by introducing randomization into its construction procedure and then making an ensemble out of it.
    - In this model I have taken certain parameters like I have taken “The number of base estimators in the ensemble is 10” and “ random state is 0”.
    - the accuracy using this model is 0.8369565217391305
  1. from sklearn import svm
     + The sklearn.svm module includes Support Vector Machine algorithms.
     + I have used C-Support Vector Classification with random state = 42.
     + I have also imported cross validation model to cross validate the score.
     + I have checked the accuracy at different CV values and picked the best among them.
     + the best accuracy using this model is at CV = 10 i.e. 0.95196078
  2. from sklearn.gaussian\_process import GaussianProcessClassifier ●

Gaussian process classification (GPC) based on Laplace approximation.

* + - The implementation is based on Algorithm 3.1, 3.2, and 5.1 of Gaussian Processes for Machine Learning (GPML) by Rasmussen and Williams.
    - I have used the parameter ‘kernel’. The kernel “1.0 \* RBF(1.0)” is used as default.
      * the accuracy using this model is 0.8369565217391305
  1. from sklearn.tree import DecisionTreeClassifier
     + The goal of this classifier is to create a model that predicts

the value of a target variable by learning simple decision rules inferred from the data features.

* + - A tree can be seen as a piecewise constant approximation. ● I have also imported cross validation model to cross validate the score.
    - I have checked the accuracy at different CV values and picked the best among them.
      * the best accuracy using this model is at CV = 16 i.e. 0.9122807