2021-2022

Heart Failure Prediction

Nizar Assad

Table of Contents

[2021-2022 1](#_Toc90282181)

[Heart Failure Prediction 1](#_Toc90282182)

[Abstract 4](#_Toc90282183)

[Introduction 5](#_Toc90282184)

[Background 6](#_Toc90282185)

[Aim 6](#_Toc90282186)

[Objectives 6](#_Toc90282187)

[Heart failure prediction dataset 7](#_Toc90282188)

[Problem to be addressed 8](#_Toc90282189)

[1- Summary of the approach 8](#_Toc90282190)

[2- Data pre-processing 9](#_Toc90282191)

[Libraries needed in the study 9](#_Toc90282192)

[Importing the data 10](#_Toc90282193)

[Data exploration 10](#_Toc90282194)

[General looking on data 10](#_Toc90282195)

[Data visualization 12](#_Toc90282196)

[3- Cleaning the data: 14](#_Toc90282197)

[4- Relationship analysis: 15](#_Toc90282198)

[5- Data preprocessing: 17](#_Toc90282199)

[Splitting the target and the features 17](#_Toc90282200)

[Splitting the data into training & test data 20](#_Toc90282201)

[1- Feature scaling: 20](#_Toc90282202)

[3- Model training and evaluation 21](#_Toc90282203)

[1- Logistic regression: 21](#_Toc90282204)

[2- Decision trees: 22](#_Toc90282205)

[3- Support Vector Machine (SVM): 23](#_Toc90282206)

[4- Random forest 24](#_Toc90282207)

[5- K-Nearest neighbors 24](#_Toc90282208)

[6- Gaussian Naïve Bayes 25](#_Toc90282209)

[Evaluating the models: 25](#_Toc90282210)

[4- Results and discussion 31](#_Toc90282211)

[Referencing: 33](#_Toc90282212)

# Abstract

Heart-related diseases are the main reason for a huge number of death in the world over the last decades and is considered as the most life-threatening disease. Our project object is to detect if a person has a heart disease or not. The main goal of this research is to improve the accuracy of diagnosis and help saving human resources. For this we will conduct a research to improve heart failure prediction using the ‘heart.csv’ dataset.

K*eywords: Machine Learning, Logistic regression, Decision tree, SVM, Heart disease.*

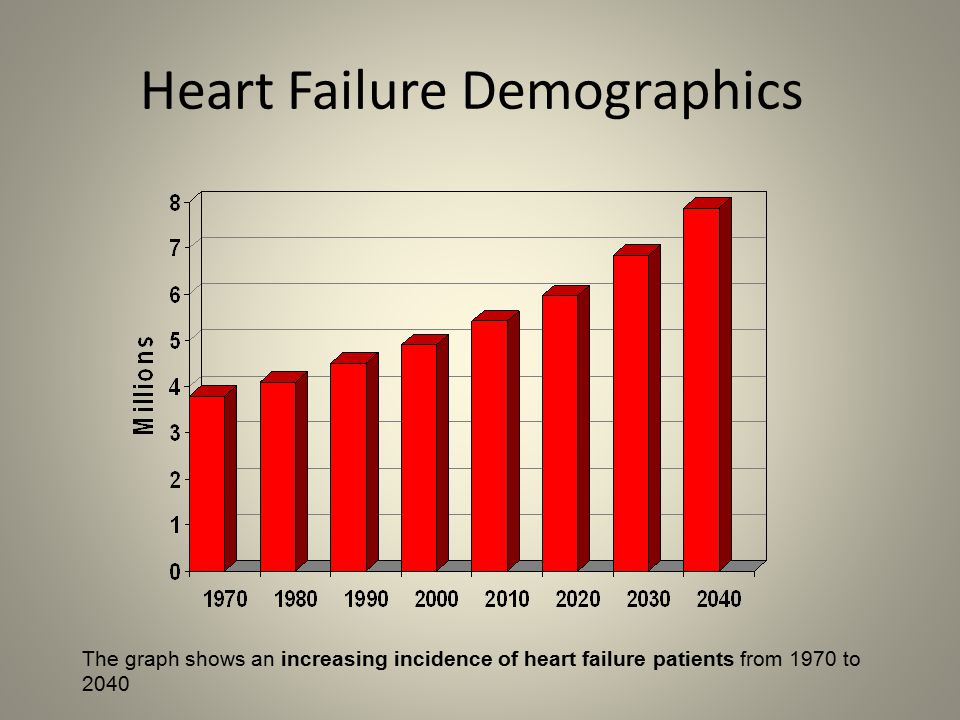
# Introduction

A heart failure, also known as congestive heart failure, happens when the heart doesn’t pump the amount of blood the body requires, which causes shortness of breath. Our problem is that we want to predict whether a person has a heart disease or not. According to the world-wide Organization, 17,9 million people die each year due to heart disease. Many deaths in all countries of the world are due to heart ailments. (Cardiovascular diseases, 2021)

To succeed in our task, the proposed model should be at the same time efficient and reliable, because the consequence are big in case the prediction is false. If a person with no heart disease is predicted with heart disease, it will have a serious impact on the patient’s mental health a cause him panic for no reason. Otherwise, if the patient has a heart disease and is predicted to be with no heart disease, he will miss the best and maybe the last chance to cure his disease. For both cases, nobody can afford any wrong diagnosis as it is very painful. If the accuracy is high enough we might encounter a large number of issues and solve all the unnecessary trouble. So, if the model is reliable, it can be applied in medical prediction.

The graph below shows the fast growth of heart failure patients from 1970 – 2040:

**Heart Failure demographics**



(Evander Holyfield, 2021)

The main goal of this data exploration and prediction is to have more insight on the health factors that causes a patient’s risk for heart disease. For this, we will explore the data, apply some Python code implementation and then start the data prediction using three different classification models that we will compare using accuracy.

# Background

In our century, people care more about their bank accounts and their net worth more than they care about their own healthcare which led to a high increase of mental health issues and physical stress. Also, a huge impact of the corona crisis which apparently causes heart suffer with muscle damage and affects heart function. (Susan Post, M.D., M.S., 2021)

Many health centers and medical organizations collect a huge amount of data about heart issues. With all this information in hand, we can apply various machine learning techniques to exploit the data and have a closer perception.

We conducted research on how people investigated heart disease prediction before conducting the studies so that we might widen our perspectives and learn from them.

Ujma Ansari used the Decision Tree model to predict heart illness in 2011 and achieved a high accuracy of 99 percent, inspiring us to utilize a superior version of Decision Tree, Random Forest. (Soni, Jyoti, 2011)

Chaitrali S. Dangare produced the forecast in 2012 using three different models, including Nave Bayes, Decision Trees, and Neural Networks. We're working with the identical data set as he did. The difference between his work and ours is that he added two more characteristics to the dataset, bringing the total number of features to 15, compared to 13 in ours. Despite the fact that there is no significant difference between 13 and 15 features in his work, what he accomplished with the dataset motivates us to make a meaningful improvement to our dataset to avoid overfitting. (Dangare, Chaitrali S., and Sulabha S. Apte, 2012)

# Aim

The aim of this project is to predicts the likelihood of patients getting heart disease, providing more knowledge. Therefore, allowing researchers to develop better ways to prevent this from happening and establish better patterns.

# Objectives

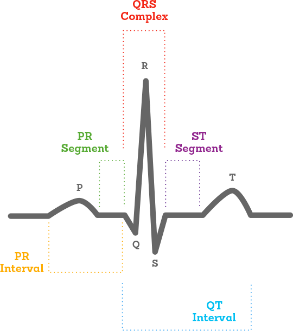
* To develop machine learning models, helping to predict possibilities of heart disease. For this, we will implement Logistic regression, Decision tree, SVM, and more classification models.
* To detect the high-risk factors which may cause a heart issue.
* To analyze different classification models applied on the ‘heart.csv’ and compare their accuracy scores.

# Heart failure prediction dataset

The Heart Failure prediction dataset selected for this project comes from the Kaggle Repository. The dataset consists of 918 patients’ data. Describing the individual’s diagnosis of heart disease. After a long research, this dataset was chosen for its results in this particular field, it offers a good compromise between efficiency and simplicity.

This data is mainly focused on analyzing the heart condition to predict the binary target, if a person is at high risk of having a heart disease or not.

All this data is based on the ECG curves which represents the functioning of a person’s heart by a cardiac rhythm.



The 12 features which represents health factors used in the project are listed below:

• Age: age in years

• Sex: (1 = male; 0 = female)

• ChestPainType: [TA, ATA, NAP, ASY].

• RestingBP: resting blood pressure (in mm Hg on admission to the hospital)

• Cholesterol: serum cholesterol in mg/dl

• FastingBS: (fasting blood sugar &gt; 120 mg/dl) (1 = true; 0 = false)

• RestingECG: resting electrocardiographic results

• MaxHR: maximum heart rate achieved

• ExerciseAngina: exercise induced angina (1 = yes; 0 = no)

• Oldpeak: ST depression induced by exercise relative to rest

• ST\_Slope: The slope of the peak exercise ST segment

• HeartDiceaser: Output column [ 1: Heart disease, 0: Normal]

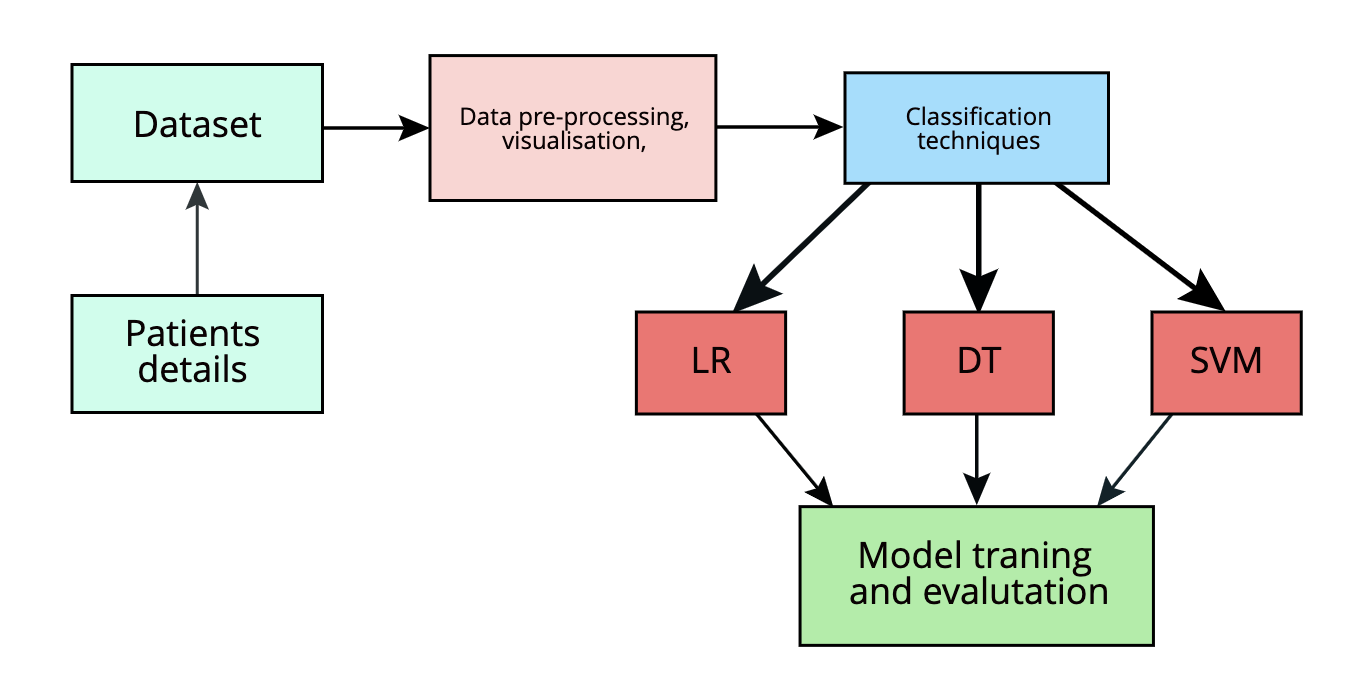
# Problem to be addressed

The major problem with heart disease is to detect it. There are many services available to check your heart and do a full diagnosis but the prices are usually too high and not 100% reliable. Also, if the disease is detected early can be very helpful as it can aid in making decisions on lifestyle changes in high risk patients which will certainty reduce the risk of a heart attack. In today’s world, we have enough data to analyze it and try to spot the most relevant risk factors and be in help fighting this condition.

**Machine learning model**

## Summary of the approach

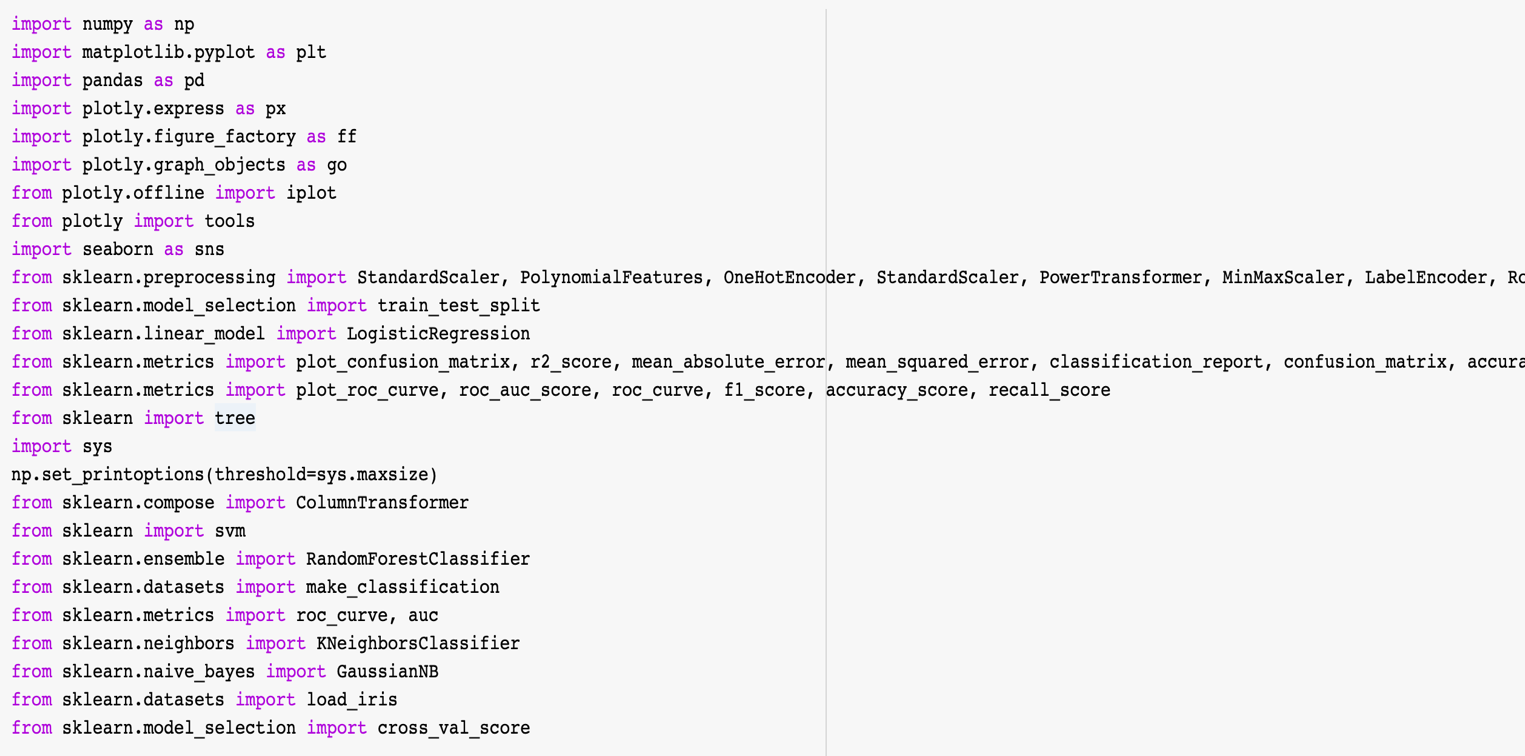
This diagram shows the steps we’re following in order to proceed with the dataset prediction.



# Data pre-processing

## Libraries needed in the study

* Obtaining data is the initial stage in the machine learning process. We employ a variety of libraries to accomplish our objectives, including Panda for data manipulation and analysis, NumPy for array operations, Matplotlib for producing aesthetically appealing and educational statistics graphics, and Sci-kit for statistical modelling. These are the librares that Data Scientists utilise the most. (I used Google Colab Notebook for this.)



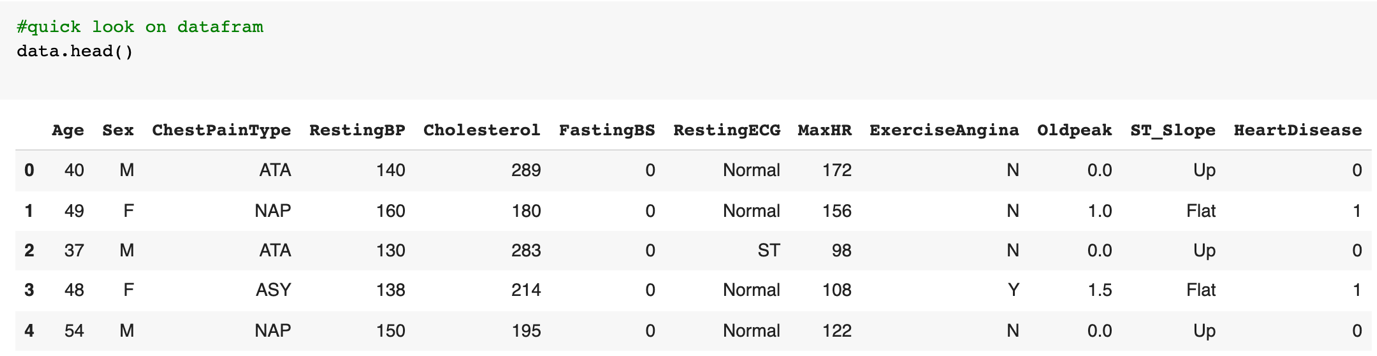
## Importing the data

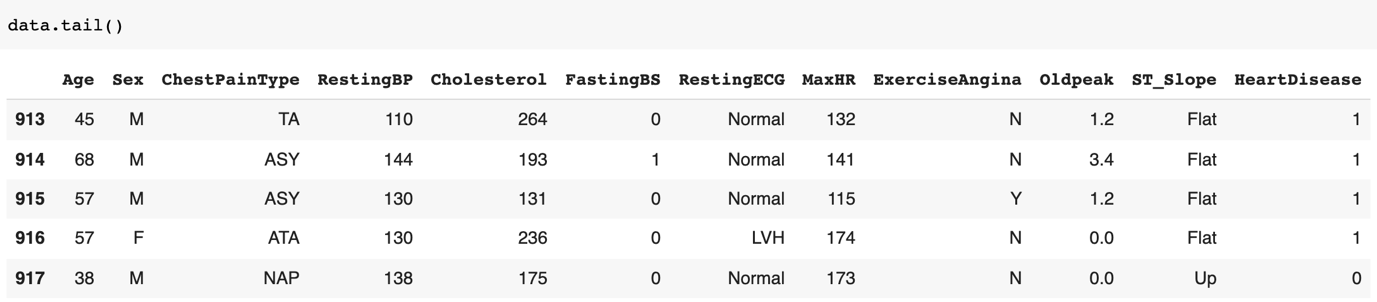
* We imported the ‘heart.csv’ dataset and name it ‘data’ using the ‘ pd ’ notation.



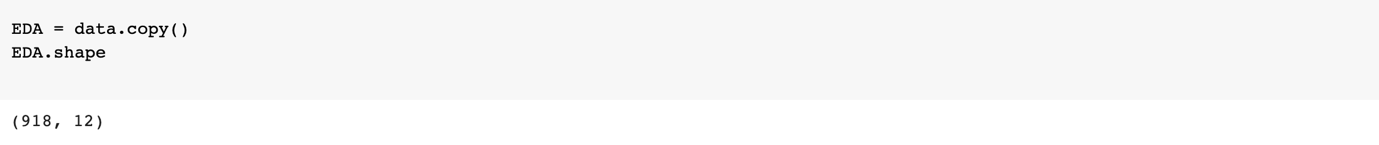
## General looking on data

* Using the head and tail functions to output the top and last 5 rows in the data set (set by default) for a quick look on data frame to get a brief idea.





* We’ve assigned EDA as a copy of data. Using the shape property to output the current shape of the array.

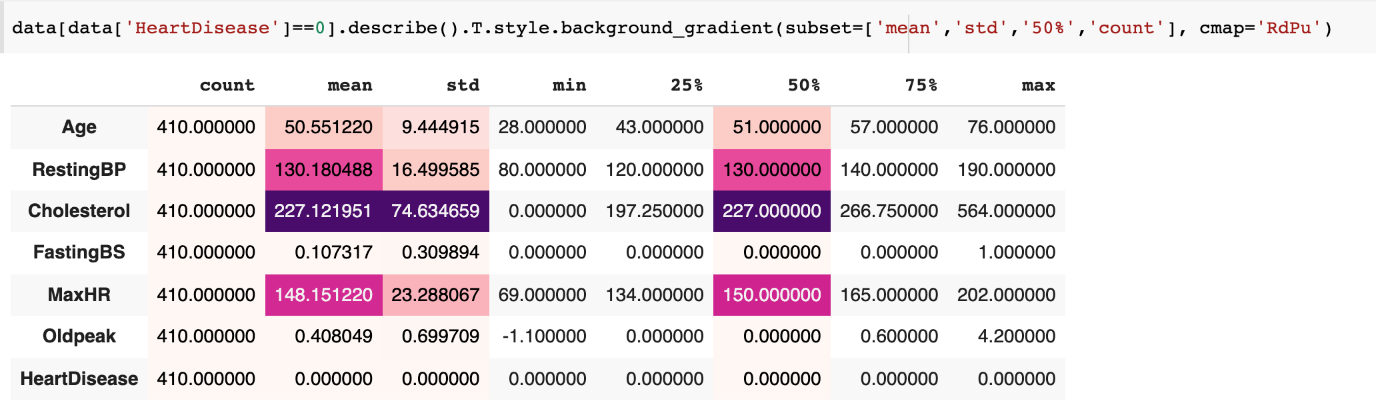


* The info() function is used to print a concise summary of a DataFrame. From this summary we can have a lot of information of data.
* Range Index.
* Number of columns.
* If there’s any null values in any of the columns.
* Type of data of every feature.
* Memory usage.

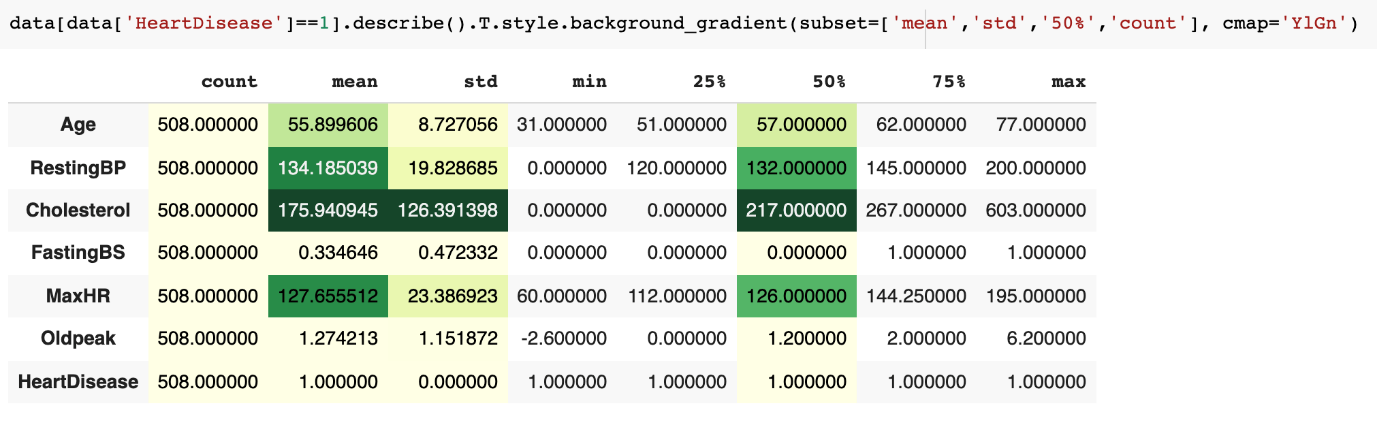


* We will use the describe function to get a summary statistic of the numerical features within the data frame. For this example, we separate the data in two different tables, one for the target value ‘0’ and a second one for the ‘1’ values.

**[‘HeartDisease’] == 0**



**[‘HeartDisease’] == 1**

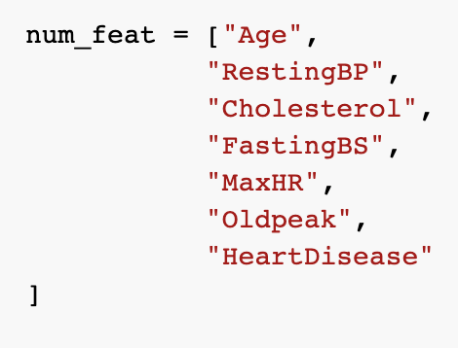


Notice the change in the color in the two tables by inputting different value to ‘cmap’

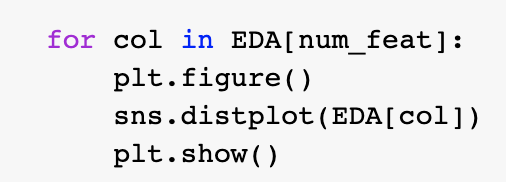
## Data visualization

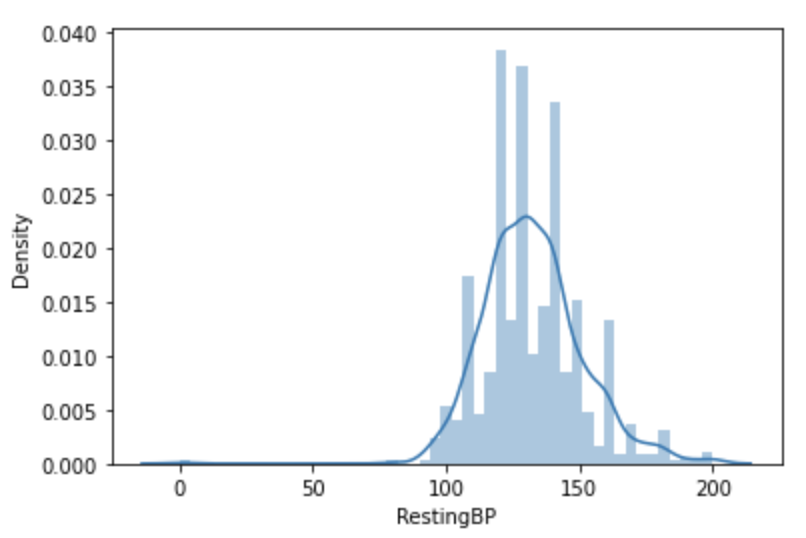
#### Numerical features:

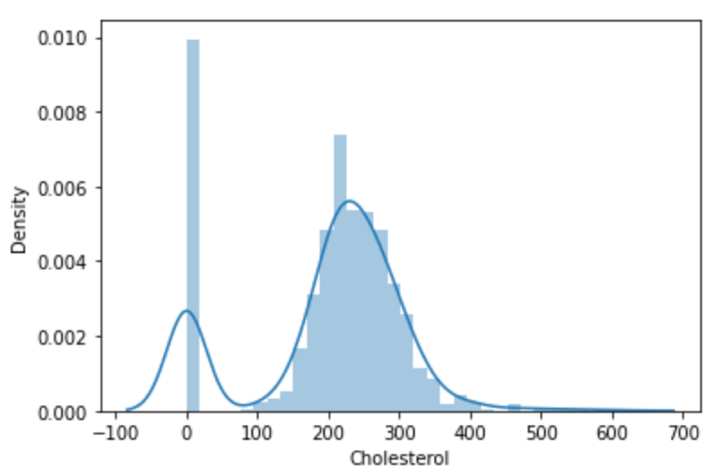
To visualize the data, we started with the numerical features. For this, we created a list names ‘num\_feat’ with all the corresponding features.

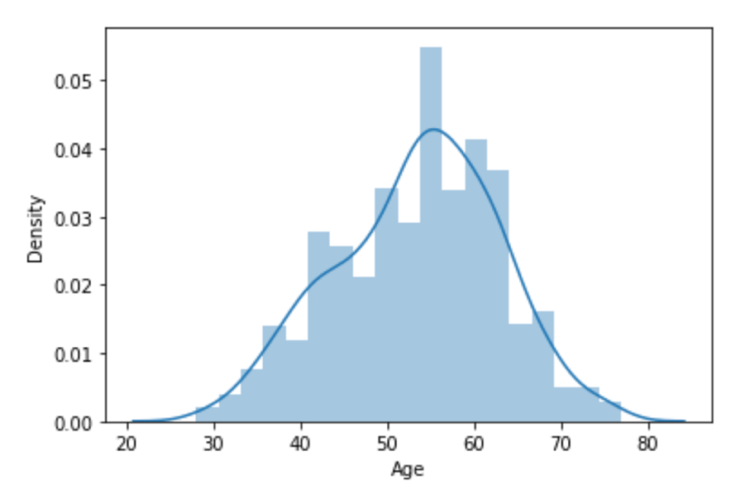


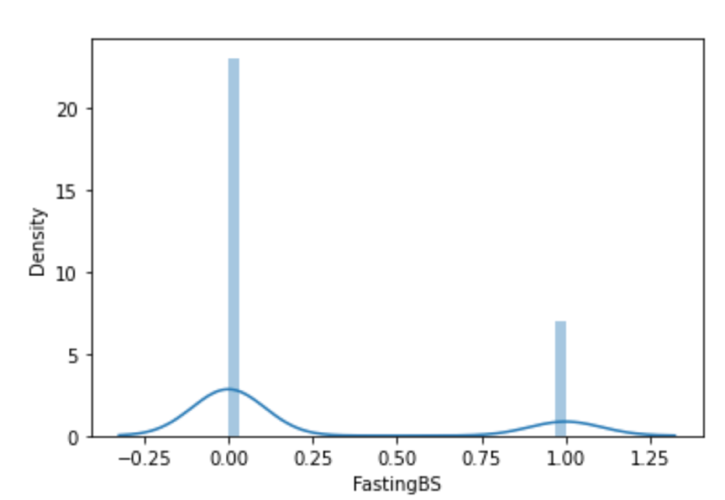
* Visualize the data with distplots using the matplotlib and seaborn libraries.

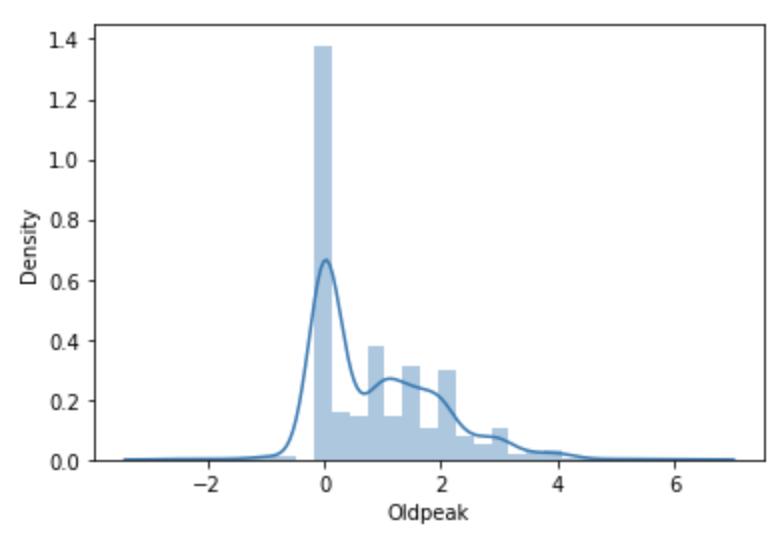


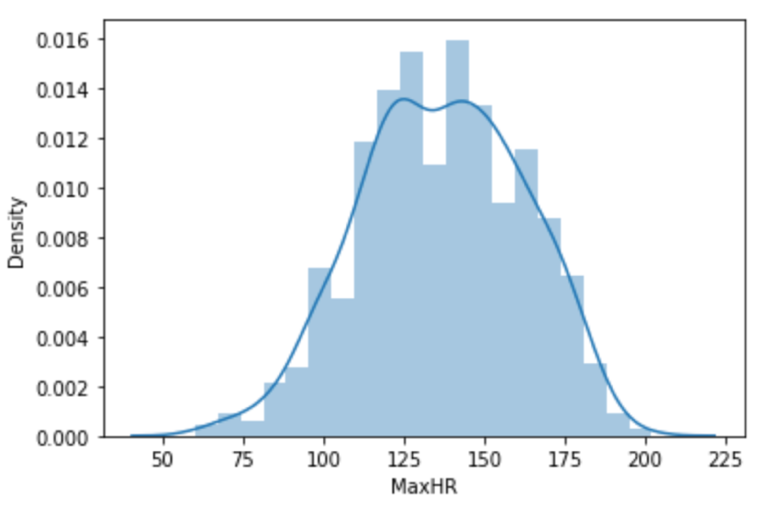


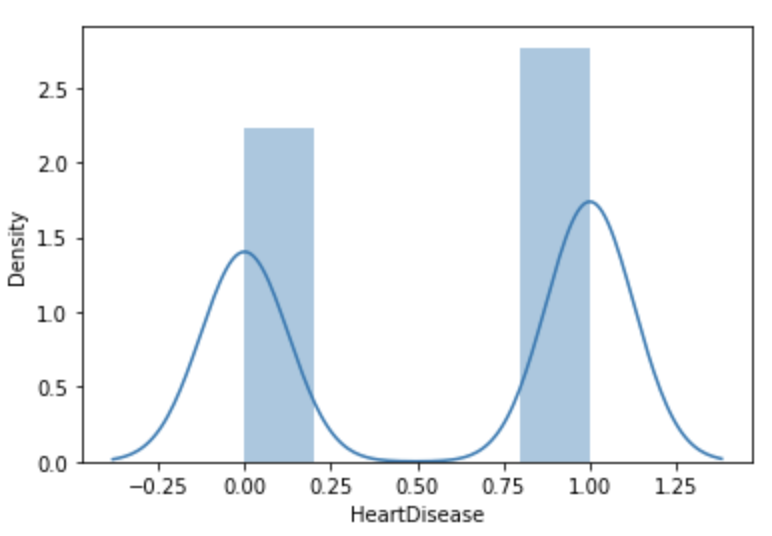






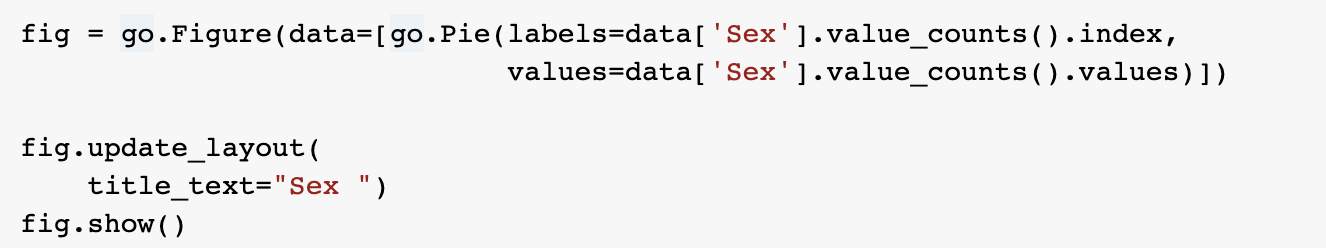


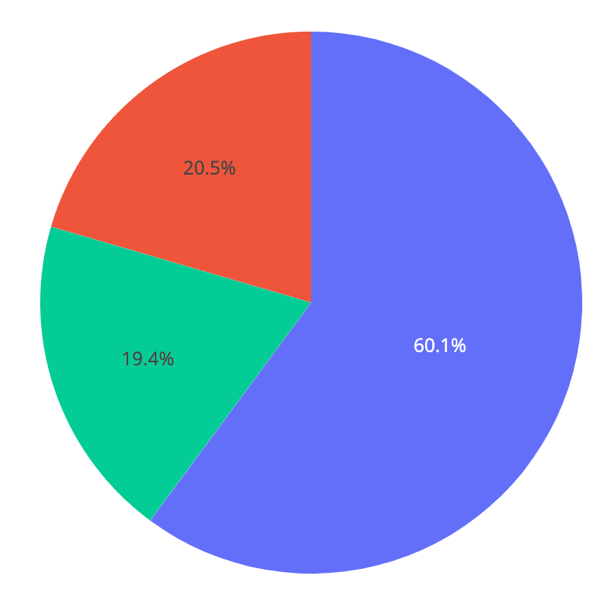


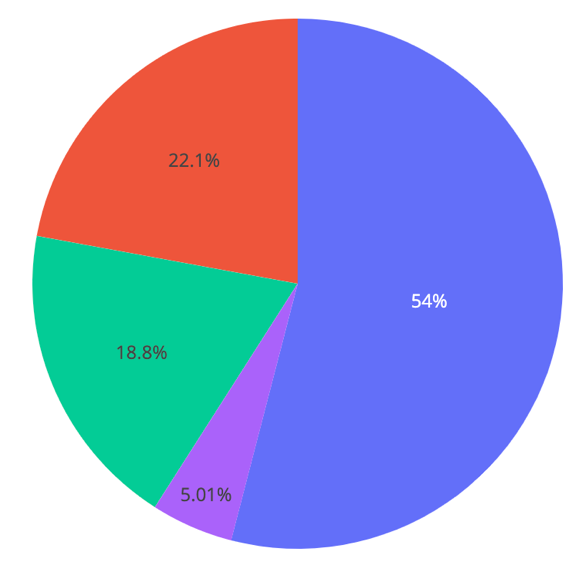


#### Categorical features:

* Circle graphs seem to be the best fit for visualizing the categorical data, we used the ‘graph\_objects’ provided with the matplotlib library.

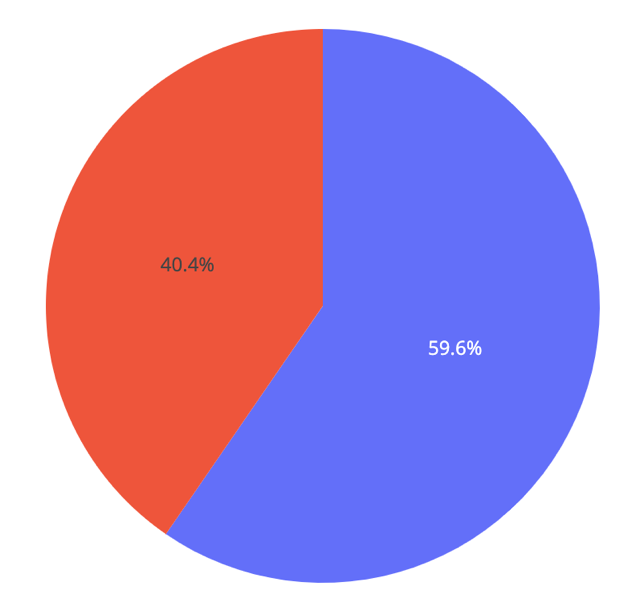


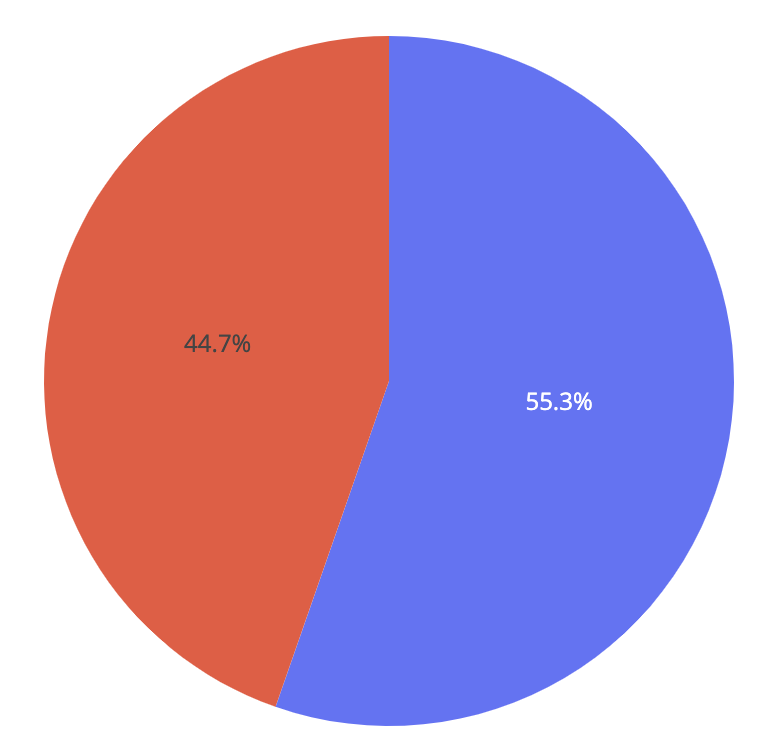


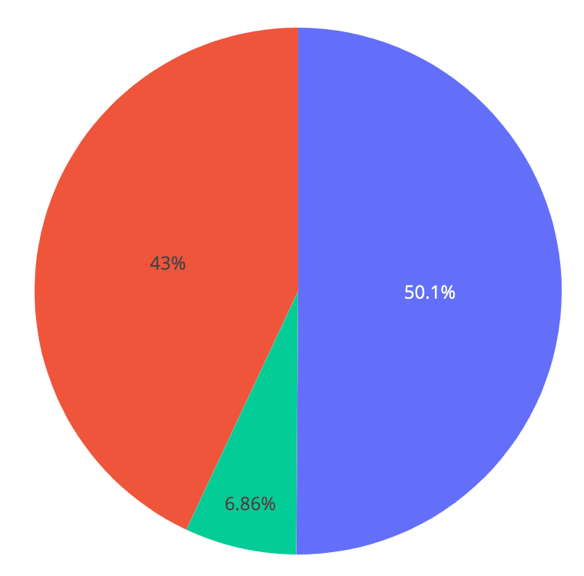




**Sex Chest pain type. RestingECG**

****





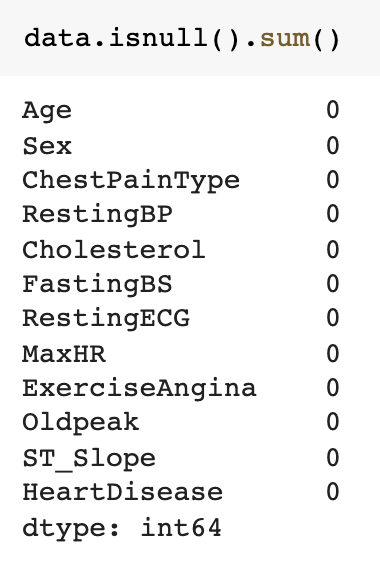
**ExerciseAngina**  **ST\_Slop Heart disease**

**Target**

Notice: The target values (heart disease feature) are balanced.

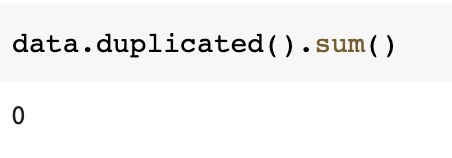
## Cleaning the data:

* Any machine learning algorithm's performance and accuracy suffer as a result of these null values. As a result, it's critical to eliminate null values from a dataset before using a machine learning algorithm on it.



There’s no missing data as we can see in the output above.

* Checking for duplicated rows is crucial for the cleaning process as it can affect the accuracy and it makes the dataset weaker and less efficient.



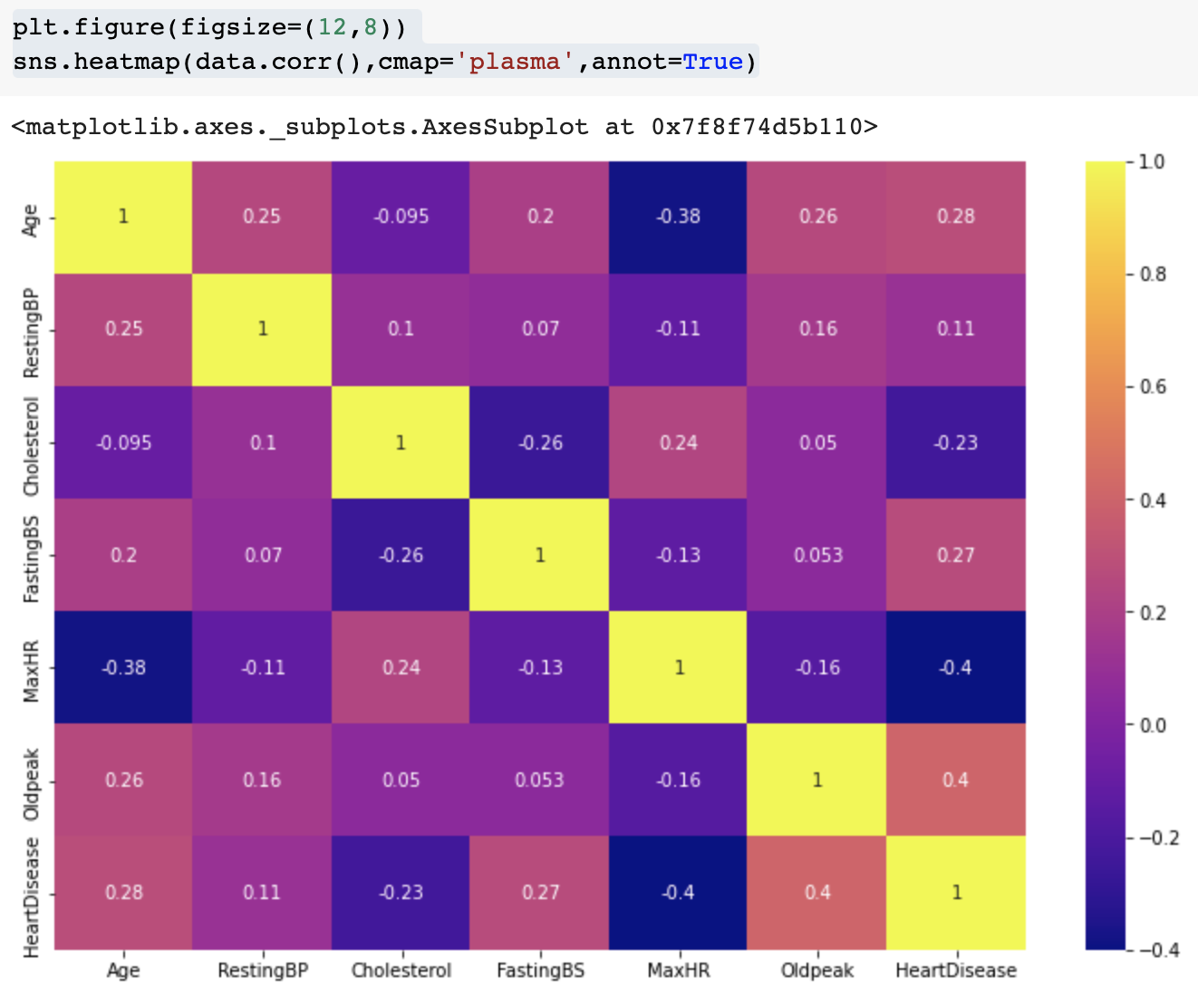
There's no duplicated rows in our dataset.

## Relationship analysis:

**Data correlation**

To check the data correlation is so important because if two variables have a linear connection, correlation can tell you how strong that relationship is.

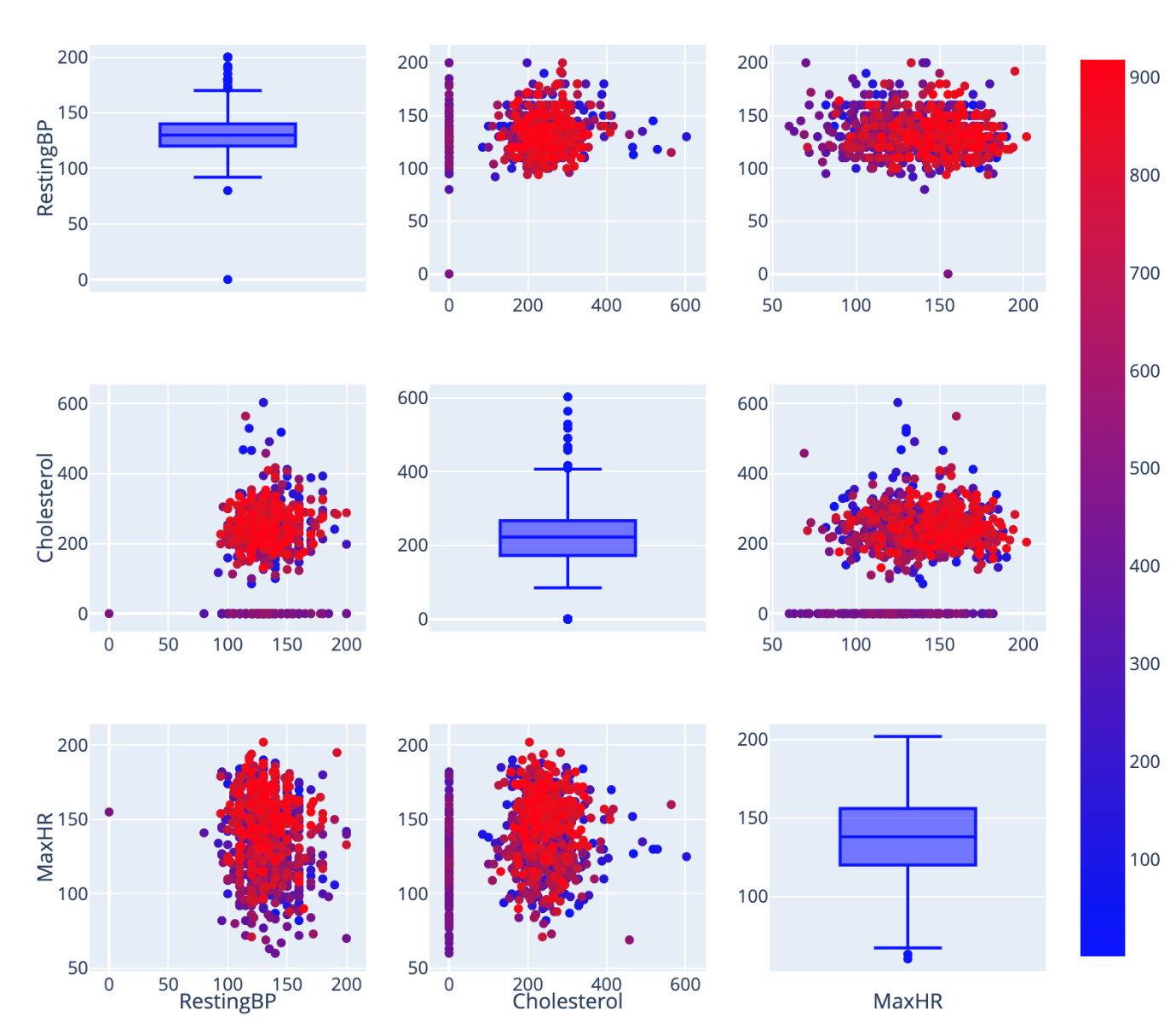
* The heatmap is used to represent correlation between features, comes with the Seaborn python library.



Notice: As we see from the heatmap, all features have a correlation to heart diseases. So, all the attributes will impact the prediction.

* Using the scatterplot matrix to visualize the correlation between the selected features. (we’ve selected these features because they have a high number of unique values)

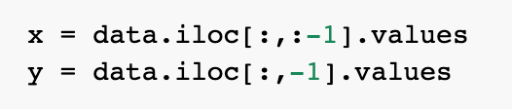




Notice: As we can see in the scatterplotmatrix, there are some outliers than needs to be removed. But after a long analysis in data, I think that all the values will be needed for this prediction as it covers all the types of patients that we could be asked to predict using the right machine learning algorithm.

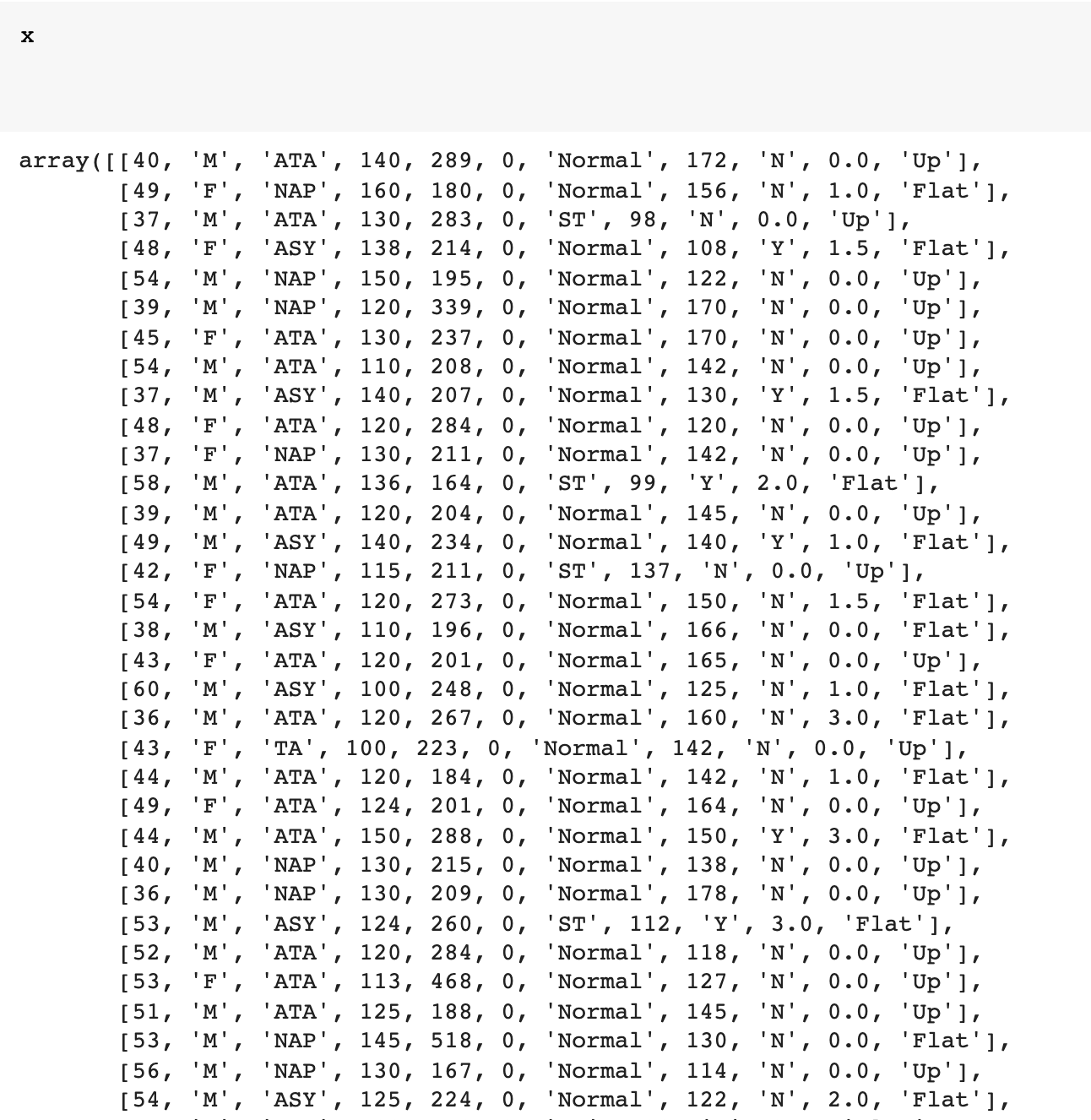
## Splitting the target and the features

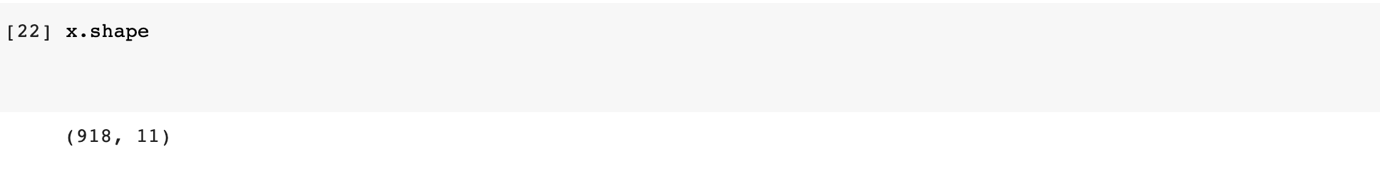
* We isolate the source variables (independent variables) from the target variable to be predicted (dependent variable) in order to simplify the next steps of data pre-processing, by adding these lines to ease the next phases of data preprocessing:



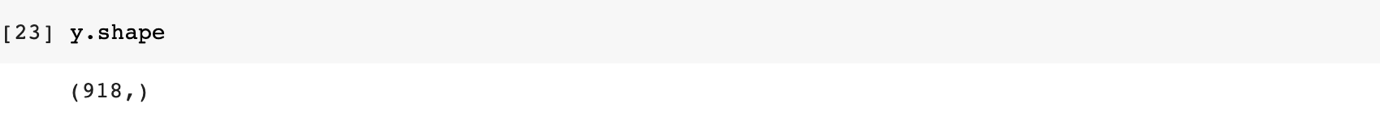
Here we called the ‘x’ array to see if the target feature separation was done successfully.

And we can that this only contain 11 columns which confirms it.

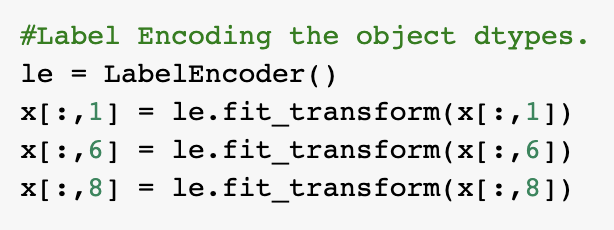
****

****

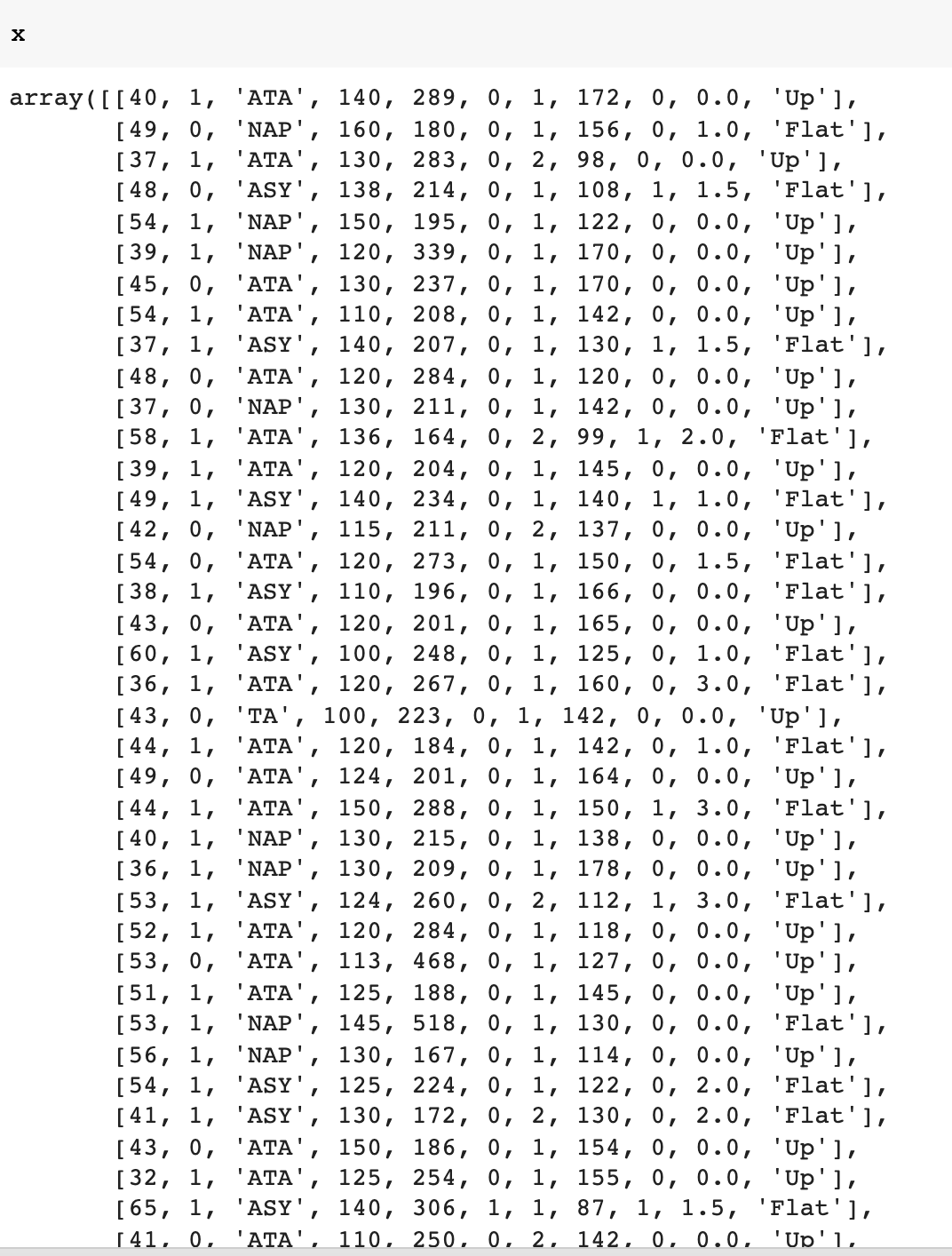
And the ‘y’ array contains one column values which represents the target feature

****

* Because machine learning models are built on mathematical equations, it's reasonable to think that keeping categorical data in the equations would cause issues because we only want numbers in the equations. That’s why we need to use the LabelEncoder function which transforms categorical features to numbers.

****

* Now let’s call the ‘x’ array to check if the encoding has succeeded.

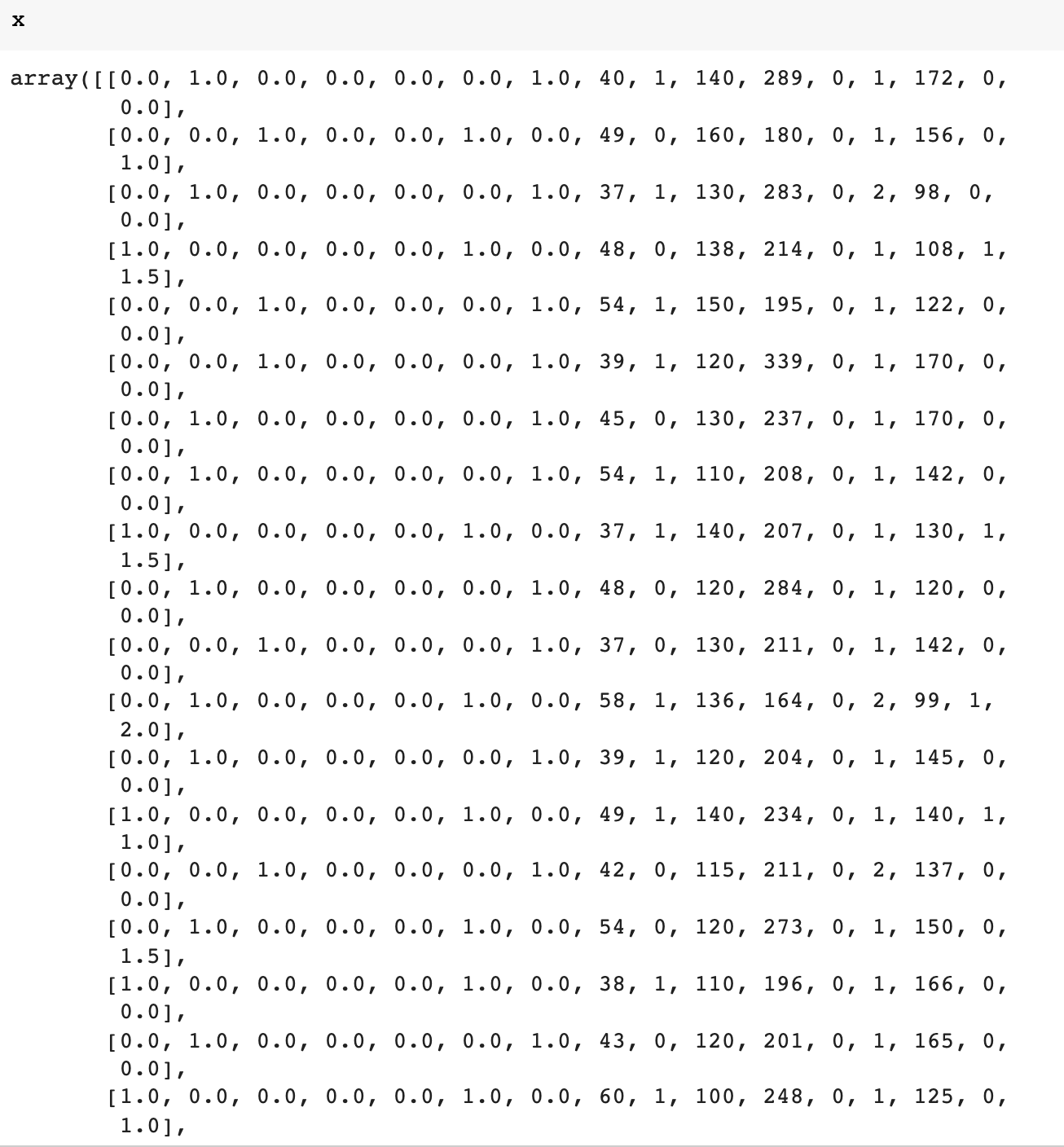
****

Notice: As we can see in the output array above, all the called features have been transformed to numerical data using the ‘fit\_transform’ function.

* Another typical approach for dealing with categorical information is one-hot encoding. It simply adds more characteristics to the category feature dependent on the number of unique values. Every category's unique value will be included as a feature.

****

Here’s the output:



As we can see in the output above, ChestPain type and ST\_Slope features have been successfully encoded and separated in features.

## Splitting the data into training & test data

After we make sure that these three points are accomplished:

* Data is clean
* Target and features are splitted.
* All features are numerical

We can start splitting data into train and test set to train our models using our dataset and then test it.

* Using the ‘train\_test\_split’ functions imported from the sklearn library. We’ve assigned 20% of the data to the test section as we have specified test\_size = 0.2. And the rest will be automatically taken by the training section.



* x\_train: contains the values of the features
* y\_train: contains the target output corresponding to x\_train values
* x\_test: contains the values of the features to be tested after training
* y\_test: contains category labels for your test data, which will be used to compare actual and projected categories.

## Feature scaling:

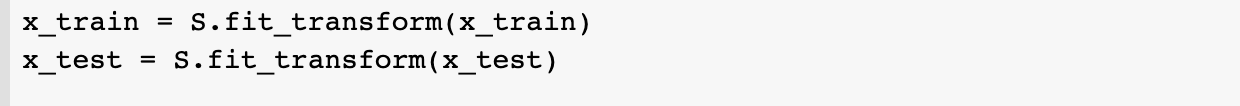
The preprocessing step of feature scaling through standardization is critical for the machine learning algorithms we're applying. Rescaling the characteristics such that they have the attributes of a conventional normal distribution with a mean of zero and a standard deviation of one is known as standardization.

`

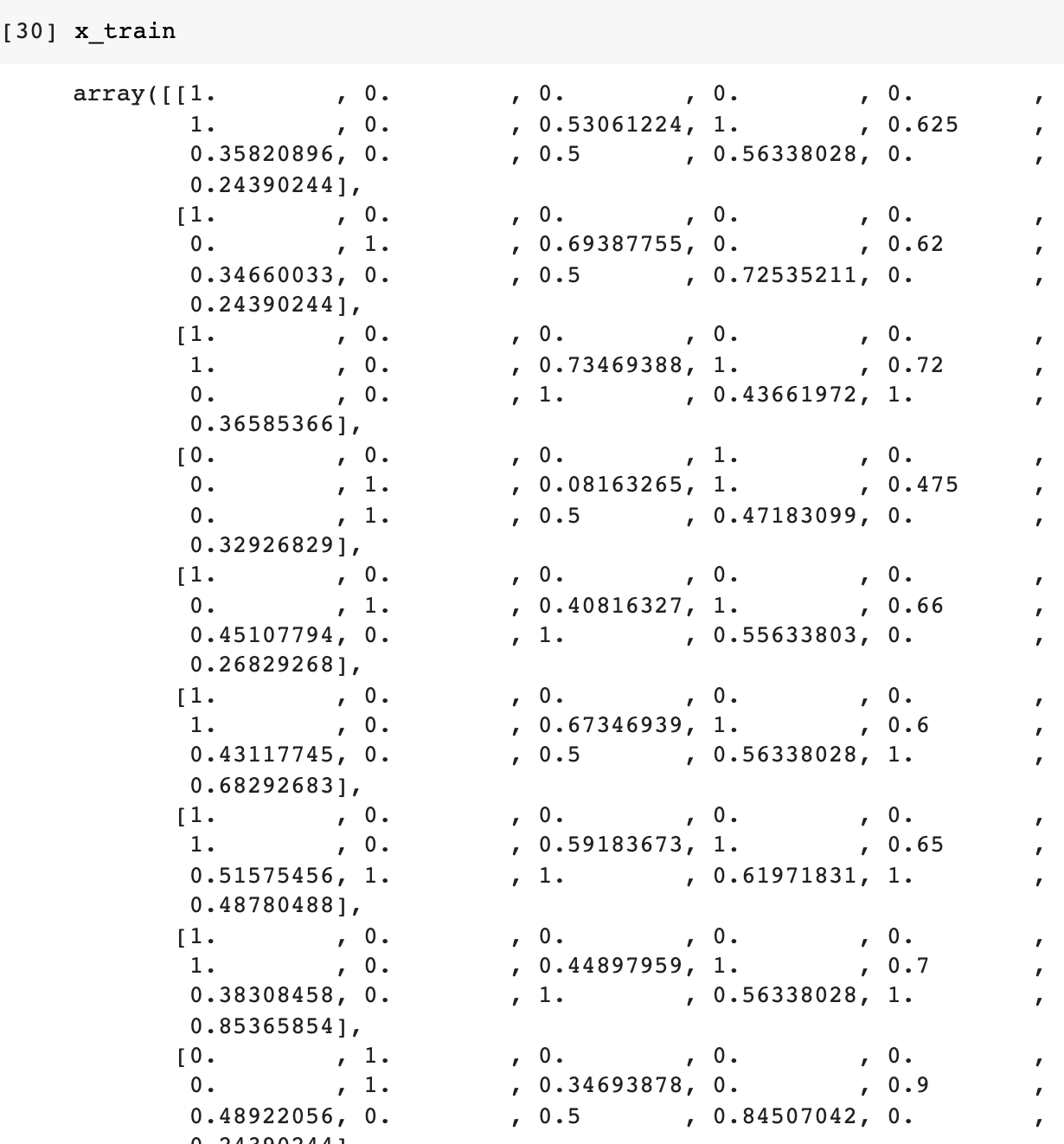
* For this, we use the ‘MinMaxScaler’ function after importing it from the sklearn library.



* We’ve assigned two variable (x\_train\_scaled, x\_test\_scaled) that will store the scaled values for the x\_train and x\_test arrays.

****

Now, let’s call ‘x\_train’ and see if the scaling has succeeded:



As we can see from the output above, the scaling has been applied to the array.

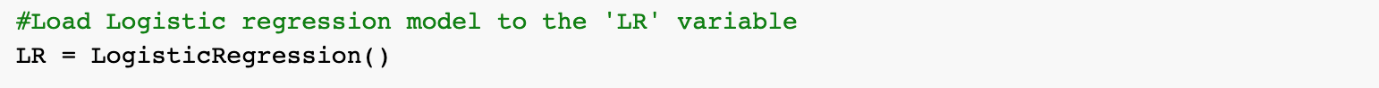
## Model training and evaluation

### Logistic regression:

The first model we decide to use is Logistic regression, it is a type of supervised learning in which the probability for classification problems with two outcomes are computed. It may also be used to predict many classes like in our example. LR\_model have been known for its high accuracy rates in prediction datasets and also for Its efficiency.

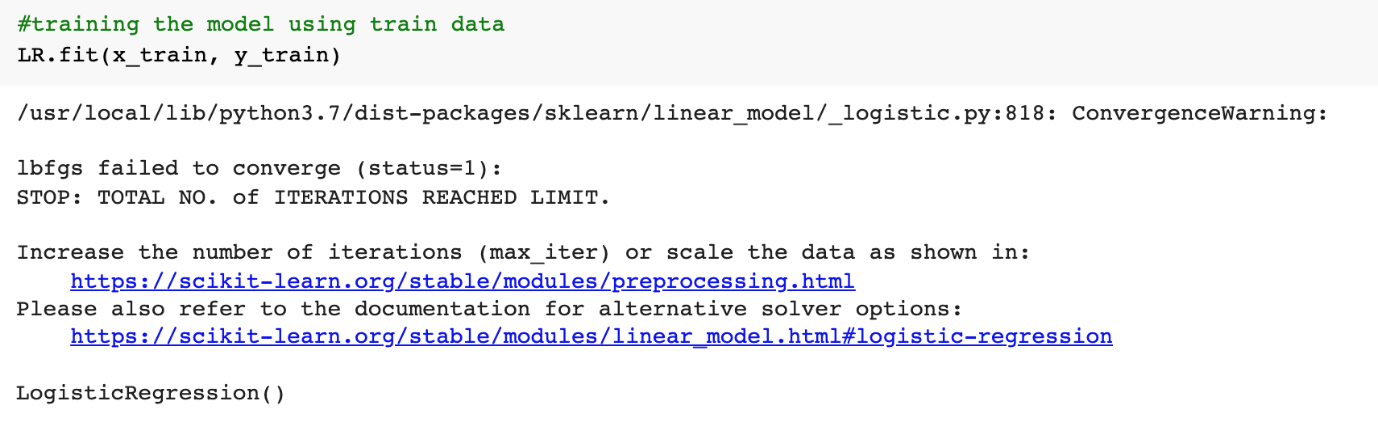
**Loading and training the model:**

* Here, we load the logistic regression into the variable that we called ‘LR’



After the model have been created, we need to fit in our training data in order to train the model.

* For this, we will use the fit method.



**Testing the model:**

After training our model, we need to test it using the testing data we’ve assigned before. We need to use the scaled features in order to get a good accuracy score.

* For this, we used the predict() function, it allows us to predict the labels of data values on the basis of the trained model.

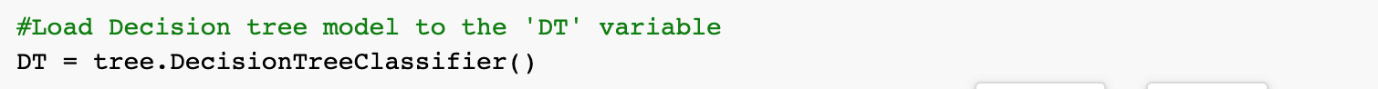


### Decision trees:

Because they are both simple to learn and effective, decision trees are often the tool of choice for predictive modelling. A decision tree's main purpose is to divide a large amount of data into smaller segments.

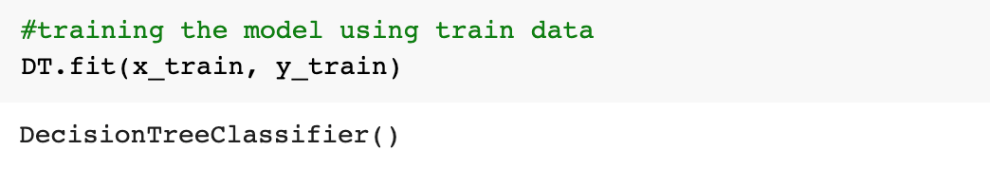
**Loading and training the model:**

* Here, we load the Decision tree model into the variable we called ‘DT’



After the model have been created, we need to fit in our training data in order to train the model.

* For this, we will use the fit method.



**Testing the model:**

Let’s test the DT model.

Using the predict function that will predict the target value based on the values captured from the ‘x\_test’ array.

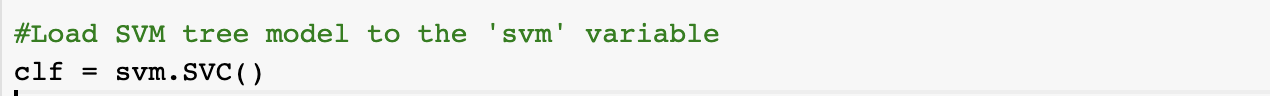


### Support Vector Machine (SVM):

SVMs (support vector machines) are a collection of supervised learning algorithms for categorization. The efficacy of support vector machines in high-dimensional areas is one of its advantages.

**Loading and training the model:**

* Here, we load the SVM model into the variable we called ‘clf’.



After the model have been created, we need to fit in our training data in order to train the model.

* For this, we will use the fit method.



**Testing the model:**

Let’s test the ‘clf’ model.

Using the predict function that will predict the target value based on the values captured from the ‘x\_test’ array.

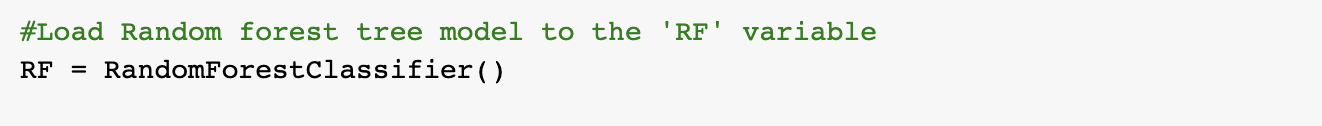


### Random forest

Random forest is a versatile, easy-to-use machine learning technique that, in most cases, gives excellent results even without hyper-parameter tuning.

**Loading and training the model:**

* Loading the Random forest model into the variable we called ‘RF’.



After the model have been created, we need to fit in our training data in order to train the model.

* Fit the data in the RF model, we will use the fit method.



**Testing the model:**

* Let’s test the ‘RF’ model.

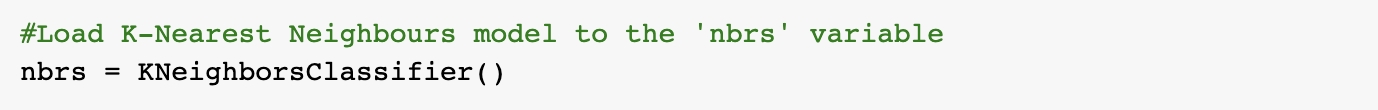
****

### K-Nearest neighbors

the k-nearest neighbors algorithm (k-NN) is a non-parametric classification method.

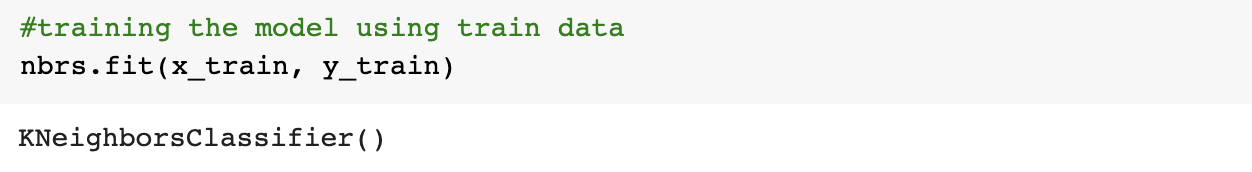
**Loading and training the model:**

* Loading the model into the variable we called ‘nbrs’



After the model have been created, we need to fit in our training data in order to train the model.

* Fitting our training data.



**Testing the model:**

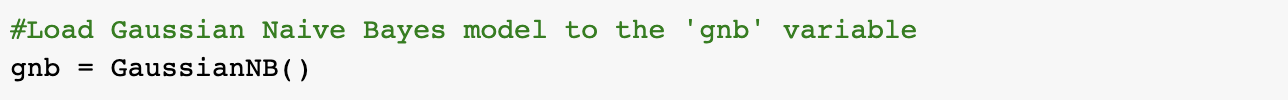


### Gaussian Naïve Bayes

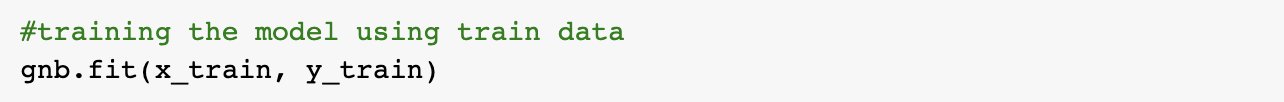
Gaussian Naive Bayes is a Naive Bayes variation that allows continuous data and follows the Gaussian normal distribution.

**Loading and training the model:**

* Loading the Gaussian Naive Bayes model into the variable called ‘gnb’.



* Fitting data.



**Testing the model:**

* Testing the ‘gnb’ model.

****

### Evaluating the models:

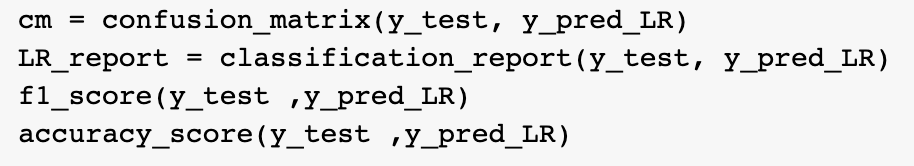
It’s essential to evaluate our models as we’re building it to find the most accurate predictions

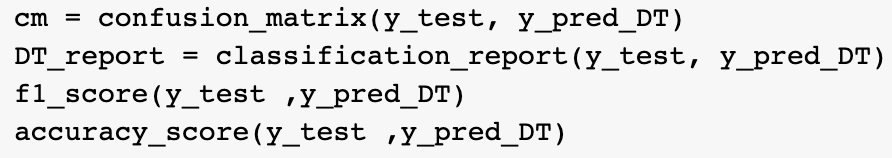
I utilized the data without applying the feature scaling to train and test the models in my initial effort to evaluate the methods functioning on this project. The findings were disappointing, therefore I employed feature scaling, which resulted in a significant change in evaluation outcomes.

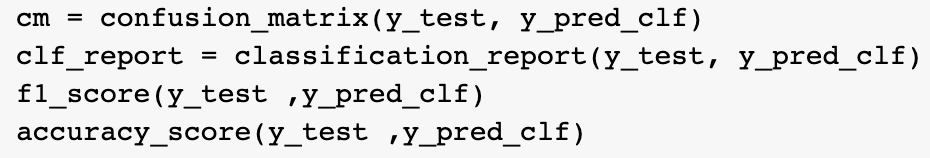
The table below show different classification metrics and confusion matrix applied to the logistic regression model before and after using feature scaling.

|  |  |
| --- | --- |
| **Before applying FS (LR)** | **After applying FS (LR)** |
| **Accuracy: 78%** | **Accuracy: 84%** |

* To evaluate and validate the models, we used accuracy, precision, recall, F1 score, and cross validation. We picked these parameters since they are the best match for our project, which is about classification.

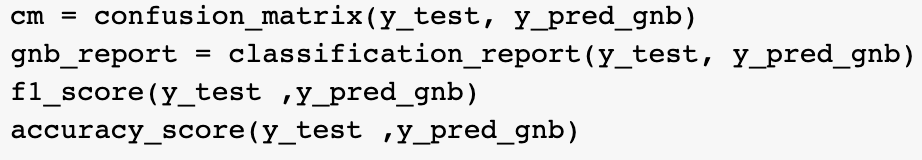


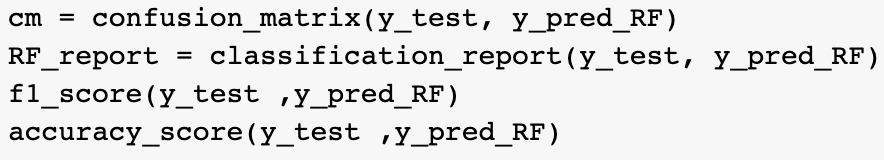




Logistic Regression Decision tree SVM

Acc: 83% Acc: 66% Acc: 84%







Random Forest K nearest neighbors Gaussian Naïve Bays

Acc: 83% Acc: 86% Acc: 84%

The K nearest neighbors has the best accuracy score of 86%.

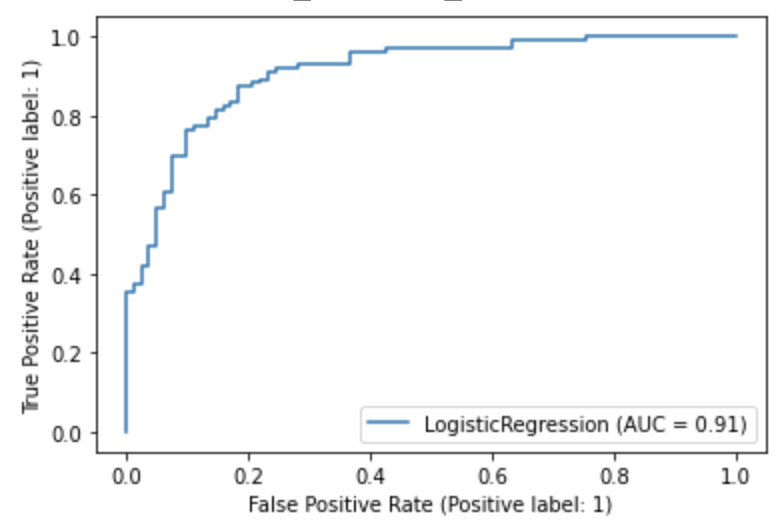
* We included all the results outputted from the sections of code above in one table for the 6 Classification methods used during this research.

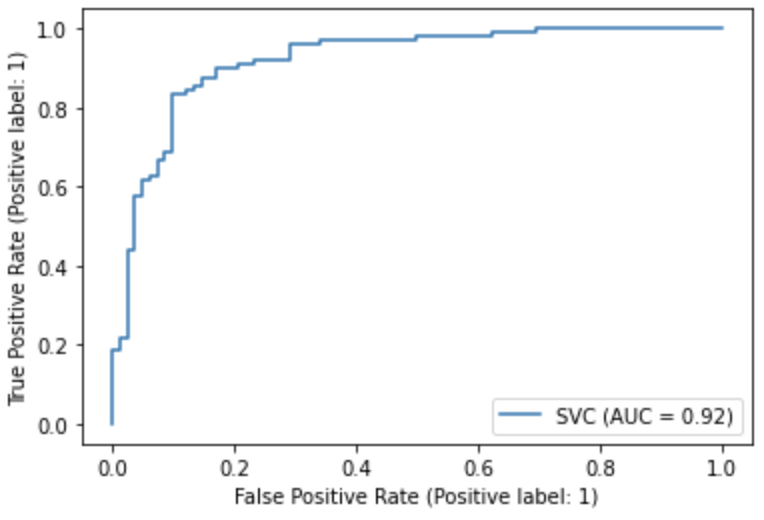
**Precision recall f1-score Cross-Val**

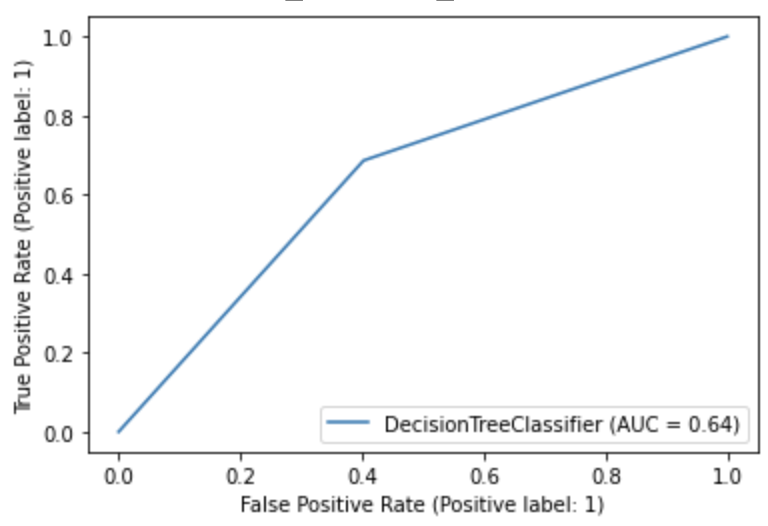
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Class 0 | Class 1 | Class 0 | Class 1 | Class 0 | Class 1 | Acc% | SD |
| Logistic Regression | 0.84 | 0.83 | 0.78 | 0.88 | 0.81 | 0.86 | 84 | 0.04 |
| Decision Tree | 0.62 | 0.70 | 0.65 | 0.68 | 0.63 | 0.69 | 75 | 0.05 |
| SVM | 0.86 | 0.82 | 0.76 | 0.90 | 0.81 | 0.86 | 83 | 0.04 |
| Random Forest | 0.89 | 0.80 | 0.71 | 0.93 | 0.79 | 0.86 | 83 | 0.04 |
| K-Nearest Neighbors | 0.88 | 0.85 | 0.79 | 0.91 | 0.83 | 0.88 | 66 | 0.05 |
| Gaussian Naive Bayes | 0.88 | 0.79 | 0.70 | 0.92 | 0.78 | 0.85 | 0.83 | 0.04 |

**ROC curves:**

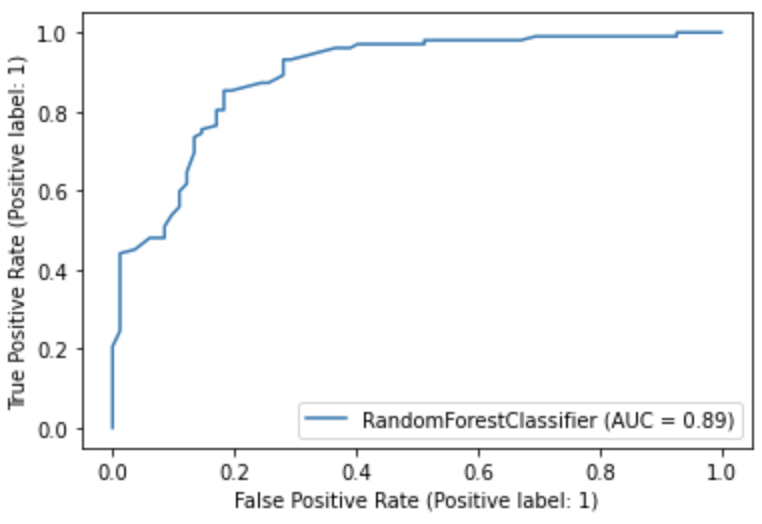
* We used ROC curves to see the performance of the trained and tested classification models at all classification thresholds.

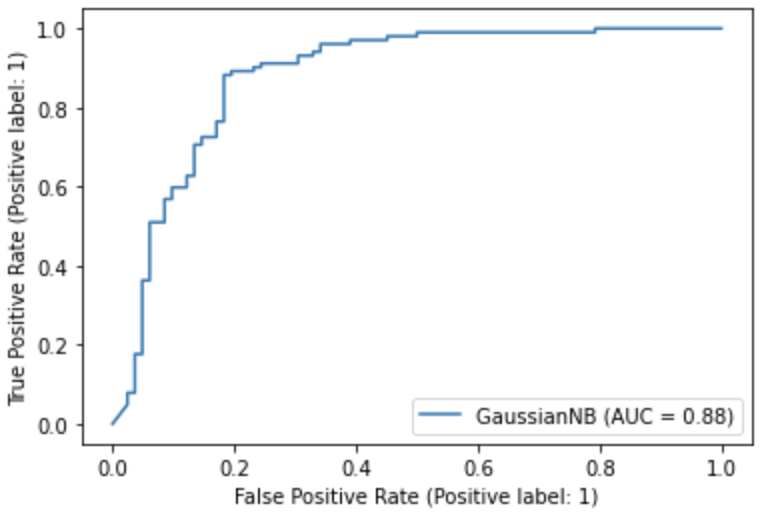


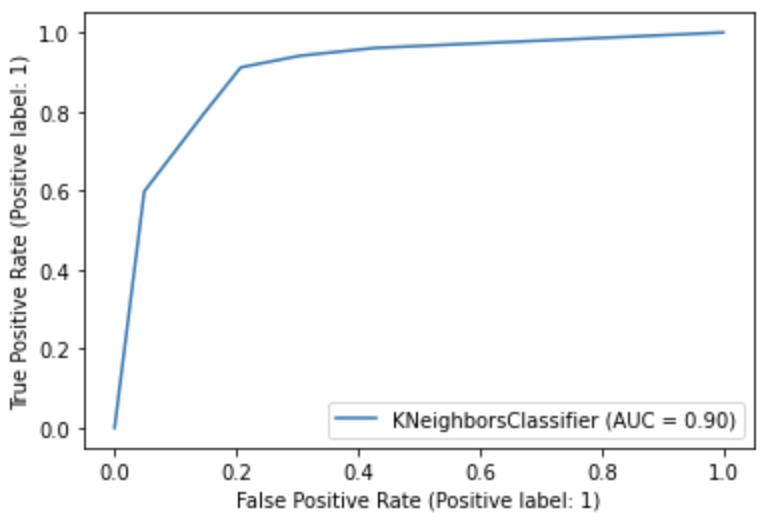




LogisticRegression DecisionTreeClassifier SVM







Random Forest K-Nearest Neighbors Gaussian Naive Bayes

As we can see in the graphs above, we can directly say that the K-Nearest Neighbors model offers the best compromise in terms of regularization.

**Confusion matrix:**

We will use confusion matrix to describe the performance of the used classification models,

It allows the visualization of the performance of an algorithm and easy identification of confusion between classes.

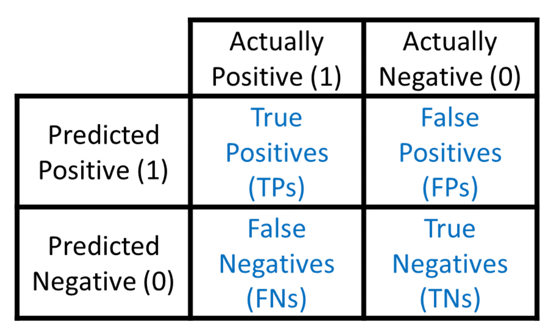
Four types of outputs can be expected when performing the classification prediction.

We'd want to go through the definitions of TP, FP, TN, and FN.

Four types of outputs can be expected when performing the classification prediction:

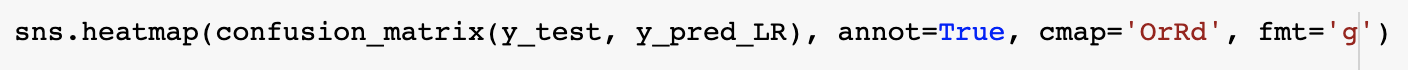
* True positive (TP)= When you predict that an observation belongs to a class, and it turns out that it does.
* False Positive (FP)= When you predict that an observation does not belong to a class, and it does not belong to that class.
* True Negative (TN)= arise when you incorrectly guess that an observation belongs to a class when it does not.
* False Negative (FN) = occur when you incorrectly guess that an observation does not belong to a class when it does.

(Evaluating a machine learning model., 2021)



(Draelos, 2019)

* For this, we’ll use the heatmap function imported from the seaborn library.
* Logistic Regression (LR):



* Decision Tree (DT):



* SVM (clf):



* Rabdom forest (RF):



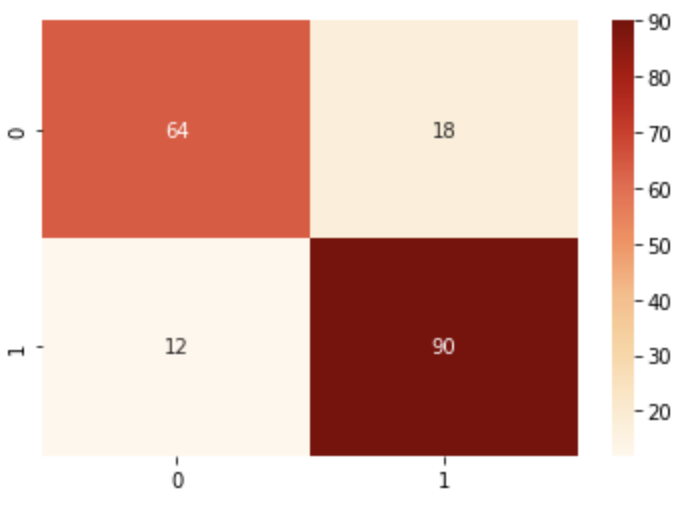
* K-nearest neighbors (nbrs):

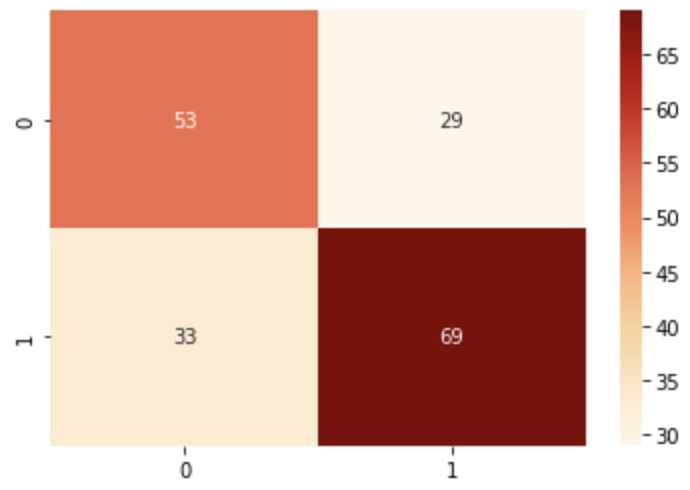


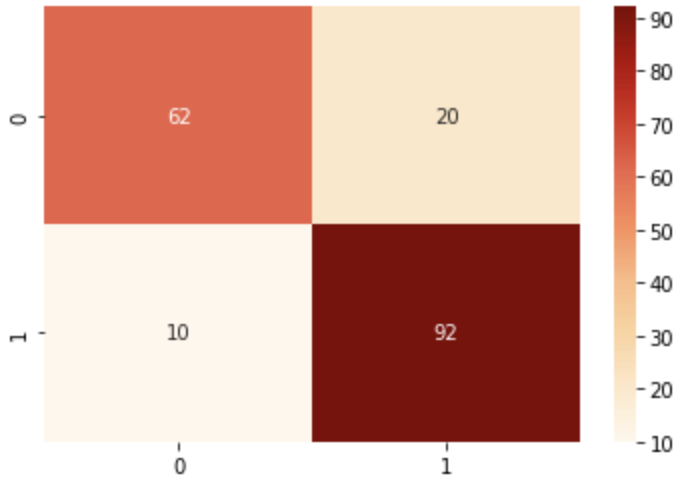
* Gaussian naïve bayes (gnb):



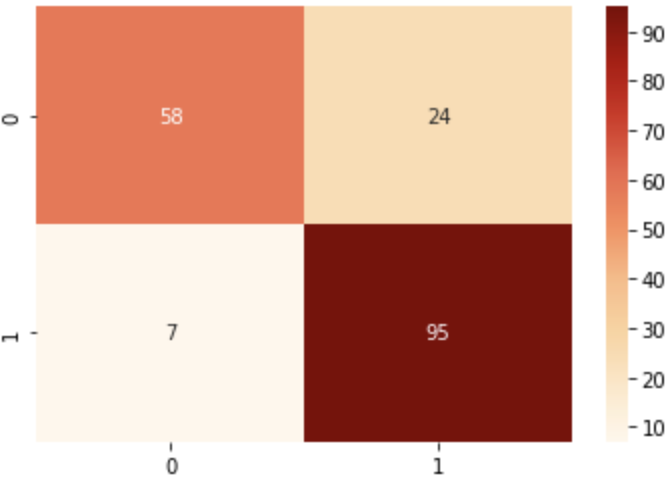
Here’s the output:

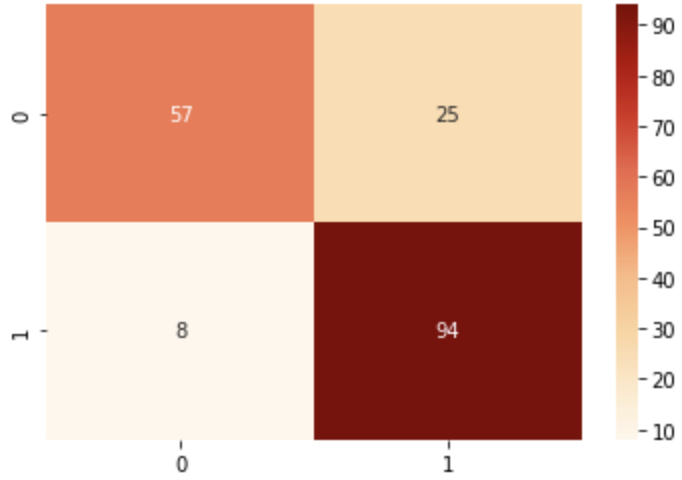






LogisticRegression DecisionTreeClassifier SVM



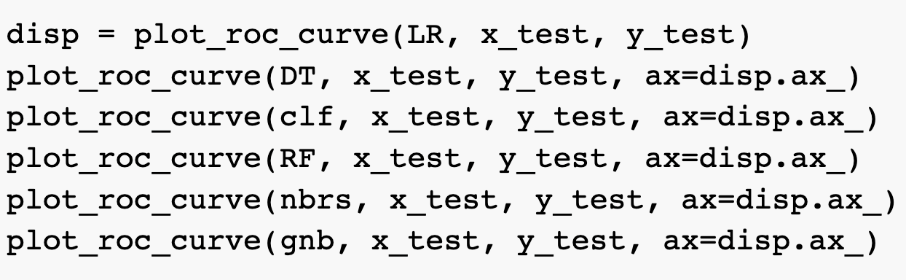




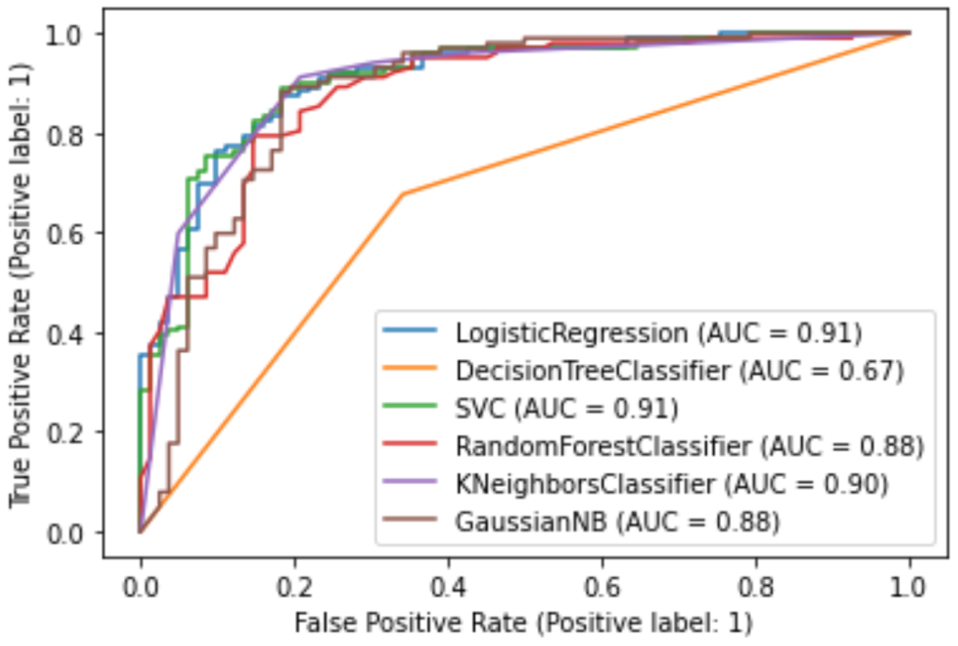
Random Forest K-Nearest Neighbors Gaussian Naive Bayes

## 4- Results and discussion

* To compare all the models, we started comparing their diagnosis tests using ROC curves.
* I used this code to display all the curves in one plot.



Output:



As we can see from the AUC results, Logitic Regression, KNeighborsClassifier are the best algorithms in terms of prediction in our health failure dataset. And from the ROC curves, we can clearly see that the area under the curve for the LR, clf, nbrs models is larger than the area under the curve for the other three models, at all cut-offs the true positive rate is higher and the false positive rate is lower than for the 3 losing models.

* After we’ve compared the ROC curves for the models, we can get to their respective confusion matrices and see what we can understand from it.

From the confusion matrices, we can see which models are doing well on the database. By looking at the TP (0,0) and TN (1,1) values, the numbers must be high and balanced.

(balanced: the gap between the two values is small). Also by looking at the FP (0,1), FN (1,0) values, the numbers must be small. In our example, K nearest neighbors has the best performance.

This project is implemented with the help of certain Python modules. Considering these metrices used to evaluate the models, we discovered that the K-Nearest Neighbors method provides the greatest test accuracy of 86 percent. It outperforms others since it is not constrained by the dataset's properties. The traits must be mutually independent according to Nave Bayes. The characteristics must be linearly separable for Logistic Regression to work. SVM necessitates the proper setting of parameters, and the neural network necessitates a complex and large dataset. Even while we receive an excellent result of 86% accuracy, it isn't adequate because it can't ensure that no incorrect diagnosis would be made. We hope to demand additional dataset in order to increase accuracy because 918 dataset instances are insufficient to conduct an amazing job. In the future, we intend to use image detection to forecast illness in diverse diseases such as lung cancer.

# Referencing:

*Susan Post, M.D., M.S., W., 2021. Heart Problems after COVID-19. [online] Hopkinsmedicine.org. Available at: <https://www.hopkinsmedicine.org/health/conditions-and-diseases/coronavirus/heart-problems-after-covid19> [Accessed 22 November 2021].*

*Evander Holyfield, L., 2021. HEARTFAILURE Living with a hurting heart - ppt download. [online] Slideplayer.com. Available at: <https://slideplayer.com/slide/5774142/> [Accessed 7 December 2021].*

Who.int. 2021. *Cardiovascular diseases*. [online] Available at: <https://www.who.int/health-topics/cardiovascular-diseases#tab=tab\_1> [Accessed 7 December 2021].

Soni, Jyoti, et al. "Predictive data mining for medical diagnosis: An overview of heart disease prediction." International Journal of Computer Applications 17.8 (2011): 43-48.

Dangare, Chaitrali S., and Sulabha S. Apte. "Improved study of heart disease prediction system using data mining classification techniques." International Journal of Computer Applications 47.10 (2012): 44-48.

Jeremy Jordan. 2021. *Evaluating a machine learning model.*. [online] Available at: <https://www.jeremyjordan.me/evaluating-a-machine-learning-model/> [Accessed 12 December 2021].

Draelos, V., 2019. *Measuring Performance: The Confusion Matrix*. [online] Glass Box. Available at: <https://glassboxmedicine.com/2019/02/17/measuring-performance-the-confusion-matrix/> [Accessed 12 December 2021].