Project: Cardiovascular Risk Prediction

Submitted By

Diksha Lonare (EBEON0123734387)

ABSTRACT

Cardiovascular disease (CVD) makes our heart and blood vessels dysfunctional and often leads to death or physical paralysis. Therefore, early and automatic detection of CVD can save many human lives. Multiple investigations have been carried out to achieve this objective, but there is still room for improvement in performance and reliability. Machine learning is the branch of Artificial Intelligence(AI), it provides prestigious support in predicting any kind of event which take training from natural events. In this paper, we calculate accuracy of machine learning algorithms for predicting cardiovascular disease, for this algorithms are k-nearest neighbour ,Logistic Regression, support vectomachine(SVM), Random Forest and XGBoost by using Kaggle cardiovascular dataset. For implementation of Python programming Anaconda(jupyter) notebook is best tool, which have many type of library, header file, that make the work more accurate and precise.

Table of Contents

CHAPTER	PARTICULARS	PAGE NO:
CHAPTER 1	Introductions: Scenario & Goals, Features & Predictor	4
CHAPTER 2	Features & Predictor	6
CHAPTER 3	Methodology: (i).Datasets (ii).Data Cleaning Preprocessing (iii). Machine Learning Algorithms (iv).Implementation Steps	7
CHAPTER 4	Analysis of the Result	27
CHAPTER 5	Conclusions	28
CHAPTER 6	Reference	29

Chapter I

INTRODUCTION

Health is a crucial part of everyone's life. However, owing to multiple reasons like unhealthy lifestyles, work stress, psychological strain, and external factors such as pollution, hazardous work environment, and lack of proper health services, millions of people worldwide fall prey to chronic ailments like cardiovascular diseases (CVD), which affect both the heart and blood vessels, resulting in death or disability. In recent years, it was reported that the majority of human deaths were due to CVD . The associated conditions are hypertension, thromboembolism, hyperlipidaemia, and coronary heart disease, which culminate in heart failure. Hypertension is the primary cause of CVD . In 2012, 7.4 million people were reported to have died from coronary heart disease, while 6.7 million people died from stroke . The World health Organization estimates that nearly 17 million people die every year from CVDs, which accounts for approximately 31% of global deaths. Early diagnosis of CVD can potentially cure patients and save innumerable lives. Diagnosis and treatment of patients at early stages by cardiologists remain a challenge. Every traditional CVD risk-assessment model implicitly assumes each risk factor related to CVD outcome in a linear fashion. Such models have a tendency to oversimplify complex relationships, including several risk factors with non-linear interactions. Multiple risk factors should be properly incorporated, and more correlated nuances between the risk factors and outcomes should be determined. To date, no large-scale study has used routine clinical data and machine learning (ML) in prognostic CVD assessment. The goal of this study is to determine if ML can enhance cardiovascular prediction accuracy in population primary care at large and find out which ML algorithm result had fairly high brevity. It includes a correlation study of categorical and continuous features of patients. In addition, data visualization and scatter plots for pairs of important features were obtained to understand the significance of the correlation between important features. These are discussed and analysed in the results section.

Scenario:

you have just been hired as a Data Scientist at a Hospital with an alarming number of patients coming in reporting various cardiac symptoms. A

cardiologist measures vitals & hands you this data to perform Data Analysis and predict whether certain patients have Heart Disease. We would like to make a Machine Learning algorithm where we can train our AI to learn & improve from experience. Thus, we would want to classify patients as either positive or negative for Heart Disease.

Goal:

• Predict whether a patient should be diagnosed with Coronary Heart Disease. This is a binary outcome.

Positive (+) = 1, patient diagnosed with Coronary Heart Disease

Negative (-) = 0, patient not diagnosed with Coronary Heart Disease

● Experiment with various Classification Models & see which yields greatest accuracy.

● Examine trends & correlations within our data

● Determine which features are most important to Positive/Negative Coronary Heart Disease diagnosis.

Chapter II

Features & Predictor

- Sex: male or female("M" or "F")
- Age: Age of the patient;
- is_smoking: whether or not the patient is a current smoker ("YES" or "NO")
- Cigs Per Day: the number of cigarettes that the person smoked on average in one day
- BP Meds: whether or not the patient was on blood pressure medication
- Prevalent Stroke: whether or not the patient had previously had a stroke
- Prevalent Hyp: whether or not the patient was hypertensive
- Diabetes: whether or not the patient had diabetes
- Tot Chol: total cholesterol level
- Sys BP: systolic blood pressure
- Dia BP: diastolic blood pressure
- BMI: Body Mass Index
- Heart Rate: heart rate
- Glucose: glucose level

Chapter:III

Methodology

Data Cleaning and Preprocessing: The datasets which were collected from UCI machine learning repository and Kaggle website contain unfiltered data which must be filtered before the final data set can be used to train the model. Also, data has some categorical variables which must be modified into numerical values for which we used Pandas library of Python. In data cleaning step, first we checked whether there are any missing or junk values in the dataset for which we used the isnull() function. Then for handling categorical variables we converted them into numerical variables.

Machine Learning Algorithms:

Logistic Regression : Logistic regression is often used a lot of times in machine learning for predicting the likelihood of response attributes when a set of explanatory independent attributes are given. It is used when the target attribute is also known as a dependent variable having categorical values like yes/no or true/false, etc. It's widely used for solving classification problems. It falls under the category of supervised machine learning. It efficiently solves linear and 12 binary classification problems. It is one of the most commonly used and easy to implement algorithms. It's a statistical technique to predict classes which are binary. When the target variable has two possible classes in that case it predicts the likelihood of occurrence of the event. In our dataset the target variable is categorical as it has only two classes-yes/no.

Decision Tree Classifier: Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

Decision trees are upside down which means the root is at the top and then this root is split into various several nodes. Decision trees are nothing but a bunch of if-else statements in layman terms. It checks if the condition is true and if it is then it goes to the next node attached to that decision.

In a Decision Tree diagram, we have:

Root Node: The first split which decides the entire population or sample data should further get divided into two or more homogeneous sets. In our case, the Outlook node.

Splitting: It is a process of dividing a node into two or more sub-nodes.

Decision Node: This node decides whether/when a sub-node splits into further sub-nodes or not. Here we have, Outlook node, Humidity node, and Windy node.

Leaf: Terminal Node that predicts the outcome (categorical or continuous value). The coloured nodes, i.e., Yes and No nodes, are the leaves.

Random Forest :

Random Forest is the most famous and it is considered as the best algorithm for machine learning. It is a supervised learning algorithm. To achieve more accurate and consistent prediction, random forest creates several decision trees and combines them together. The major benefit of using it is its ability to solve both regression and classification issues. When building each individual tree, it employs bagging and feature randomness in order to produce an uncorrelated tree forest whose collective forecast has much better accuracy than any individual tree's prediction. Bagging enhances accuracy of machine learning methods by grouping them together. In this algorithm, during the splitting of nodes it takes only random subset of nodes into an account. When splitting a node, it looks for the best feature from a random group of features rather than the most significant feature. This results into getting better accuracy. It

efficiently deals with the huge datasets. It also solves the issue of overfitting in datasets. It works as follows: First, it'll select random samples from the provided dataset. Next, for every selected sample it'll create a decision tree and it'll receive a forecasted result from every created decision tree. Then for each result which was predicted, it'll perform voting and through voting it will select the best predicted result.

K Nearest Neighbor (KNN) : KNN is a supervised machine learning algorithm. It assumes similar objects are nearer to one another. When the parameters are continuous in that case knn is preferred. In this algorithm it classifies objects by predicting their nearest neighbor. It's simple and easy to implement and also has high speed because of which it is preferred over the other algorithms when it comes to solving classification problems. The algorithm classifies whether or not the patient has disease by taking the heart disease dataset as an input. It takes input parameters like age, sex, chol, etc and classify person with heart disease.

Algorithm takes following steps :-

Step 1: Select the value for K.

Step 2 : Find the Euclidean distance of K no. of neighbors.

Step 3 : Based on calculated distance, select the K nearest neighbors in the training 13 data which are nearest to unknown data points.

Step 4 : Calculate no. of data points in each category among these K neighbors.

Step 5 : Assign new data points to the category which has the maximum no. of neighbors.

Step 6 : Stop.

Implementation Steps:

As we already discussed in the methodology section about some of the implementation details. So, the language used in this project is Python programming. We're running python code in anaconda navigator's Jupyter notebook. Jupyter notebook is much faster than Python IDE tools like PyCharm or Visual studio for implementing ML algorithms. The advantage of Jupyter notebook is that while writing code, it's really helpful for Data visualization and plotting some graphs like histogram and heatmap of correlated matrices. Let's revise implementation steps :

a) Dataset collection.

b) Importing Libraries : Numpy, Pandas, Scikit-learn, Matplotlib and Seaborn libraries were used.

c) Exploratory data analysis : For getting more insights about data.

d) Data cleaning and preprocessing : Checked for null and junk values using isnull() and isna().sum() functions of python.In Preprocessing phase, we did feature engineering on our dataset. As we converted categorical variables into numerical variables using function of Pandas library. Both our datasets contains some categorical variables

e) Feature Scaling : In this step, we normalize our data by applying Standardization by using StandardScalar() and fit_transform() functions of scikit-learn library.

f) Model selection : We first separated X's from y's. X's are features or input variables of our datasets and y's are dependent or target variables which are crucial for predicting disease. Then using by the importing model_selection function of the sklearn library, we splitted our X's and y's into train and test

split using train_test_split() function of sklearn. We splitted 75% of our data for training and 25% for testing.

g) Applied ML models and created a confusion matrix of all models.

LIBRARYS

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import warnings warnings.filterwarnings('ignore')

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model

Data Wrangling

```
#Checking the first 5 rows.
df.head()
```

	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearC
0	64	2.0	F	YES	3.0	0.0	0	0	0	221.0	148.0	85.0	NaN	90.0	80.0	
1	36	4.0	M	NO	0.0	0.0	0	1	0	212.0	168.0	98.0	29.77	72.0	75.0	
2	46	1.0	F	YES	10.0	0.0	0	0	0	250.0	116.0	71.0	20.35	88.0	94.0	
3	50	1.0	M	YES	20.0	0.0	0	1	0	233.0	158.0	88.0	28.26	68.0	94.0	
4	64	1.0	F	YES	30.0	0.0	0	0	0	241.0	136.5	85.0	26.42	70.0	77.0	

```
#Checking the shape of the data.
df.shape
```

```
(3390, 16)
```

```
#Checking for the title of all the columns.
df.columns
```

```
Index(['age', 'education', 'sex', 'is_smoking', 'cigsPerDay', 'BPMeds',
       'prevalentStroke', 'prevalentHyp', 'diabetes', 'totChol', 'sysBP',
       'diaBP', 'BMI', 'heartRate', 'glucose', 'TenYearCHD'],
      dtype='object')
```

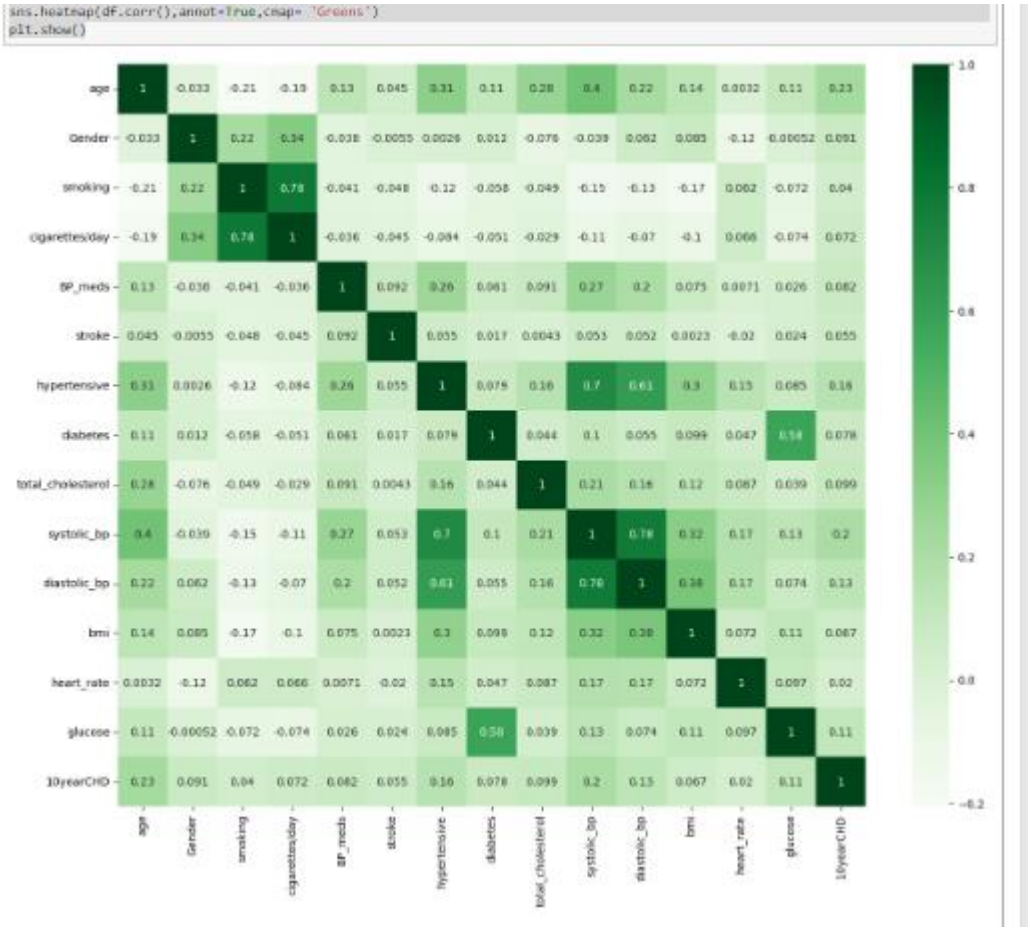
```
#Checking for null entries and data type of each column.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3390 entries, 0 to 3389
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   age              3390 non-null   int64  
 1   education        3303 non-null   float64
 2   sex              3390 non-null   object 
 3   is_smoking       3390 non-null   object 
 4   cigsPerDay       3368 non-null   float64
 5   BPMeds           3346 non-null   float64
 6   prevalentStroke  3390 non-null   int64  
 7   prevalentHyp     3390 non-null   int64  
 8   diabetes         3390 non-null   int64  
 9   totChol          3352 non-null   float64
 10  sysBP            3390 non-null   float64
 11  diaBP            3390 non-null   float64
 12  BMI              3376 non-null   float64
 13  heartRate        3389 non-null   float64
 14  glucose          3086 non-null   float64
 15  TenYearCHD       3390 non-null   int64  
dtypes: float64(9), int64(5), object(2)
memory usage: 423.9+ KB
```

Exploratory Data Analysis Correlation Matrix-

Let's you see correlations between all variables. Within seconds, you can see whether something is positively or negatively correlated with our predictor (target).

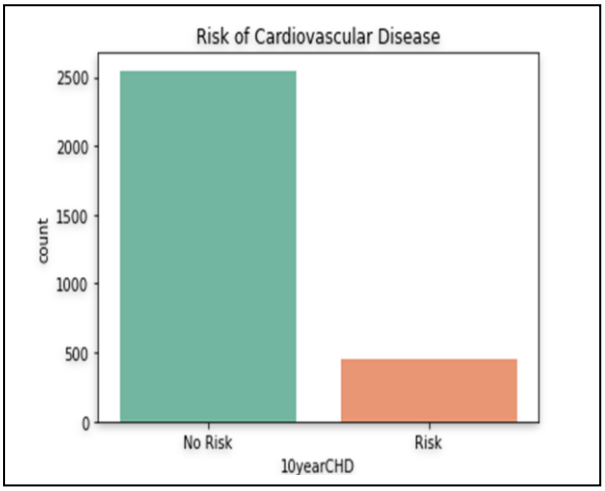
```
df.corr()

plt.figure(figsize=(15,12))

sns.heatmap(df.corr(),annot=True,cmap= 'Greens')

plt.show()
```

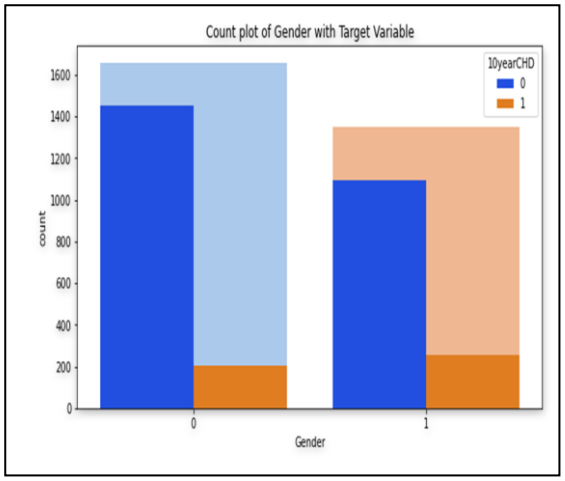


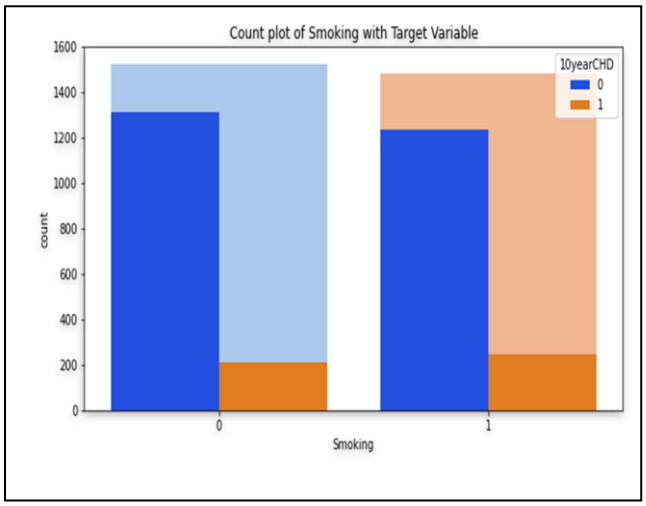
- From the above Heatmap, We can see both of these columns are heavily correleted, there's some relationship we can establish with these two features further.
- Also Elevation of systolic blood pressure predicts the risk of cardiovascular disease better than increases in diastolic blood pressure. Although associated with more variability in measurement, systolic blood pressure is easier to determine and allows more appropriate risk stratification than diastolic blood pressure.

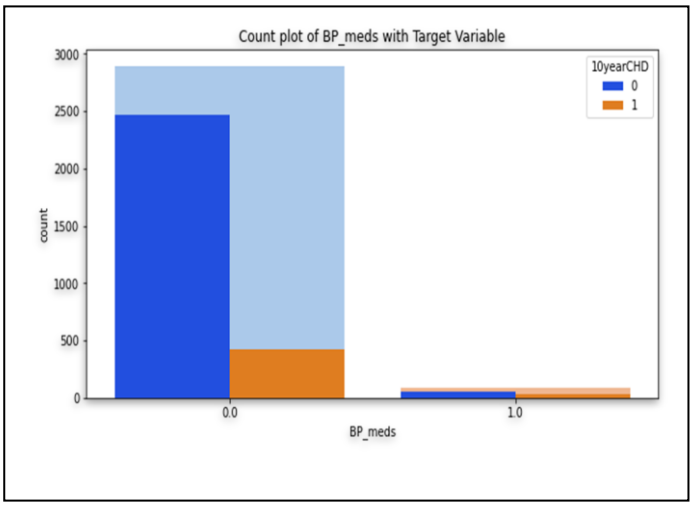
Visualizing The Distribution

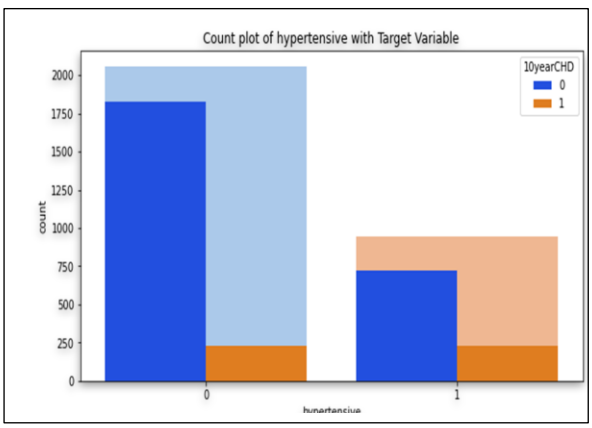
Univariate Analysis of Categorical Features & relation with target variable

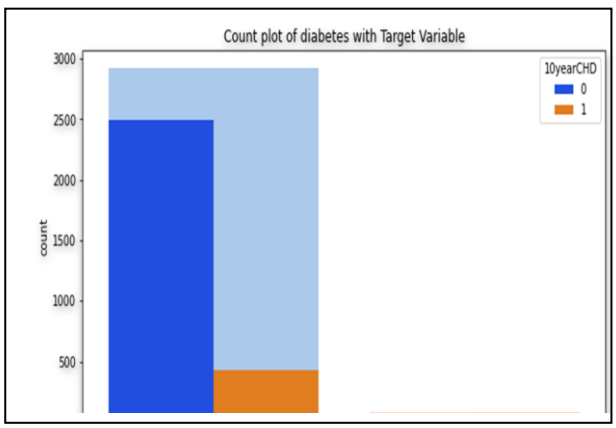




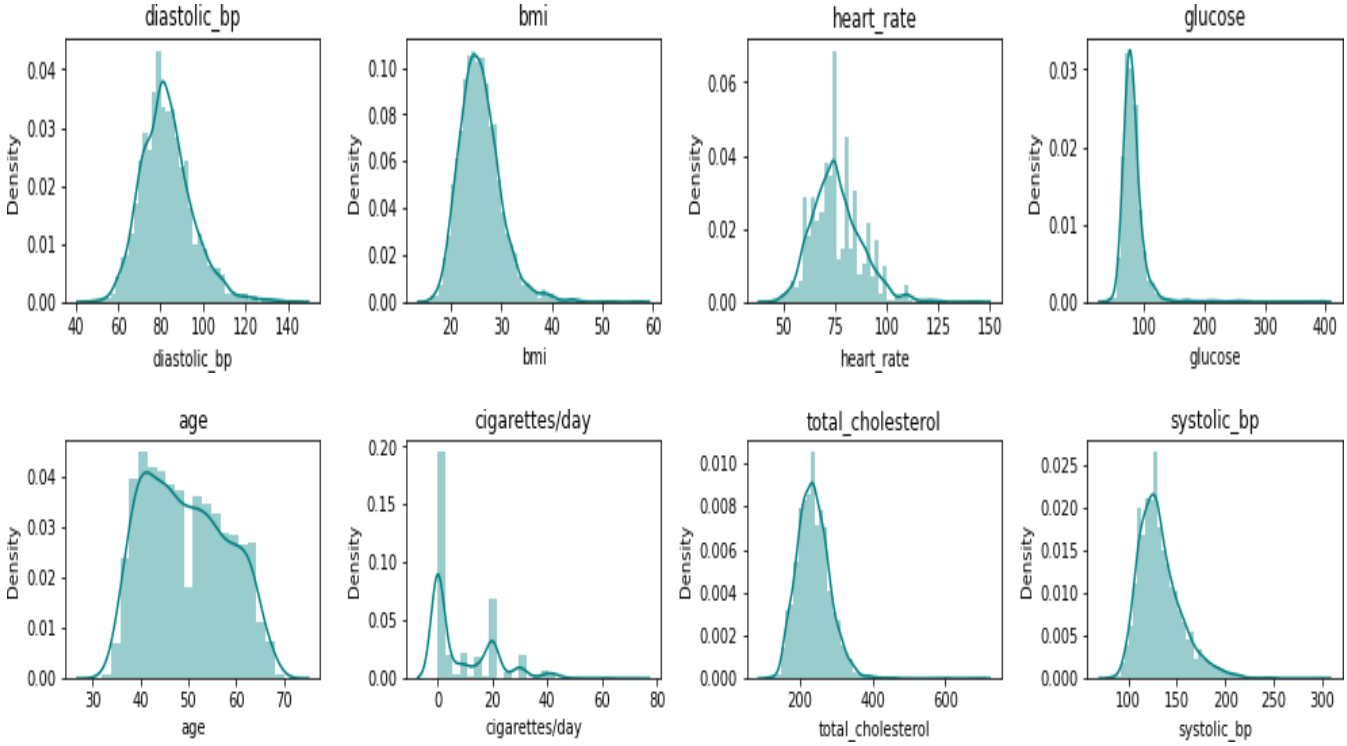








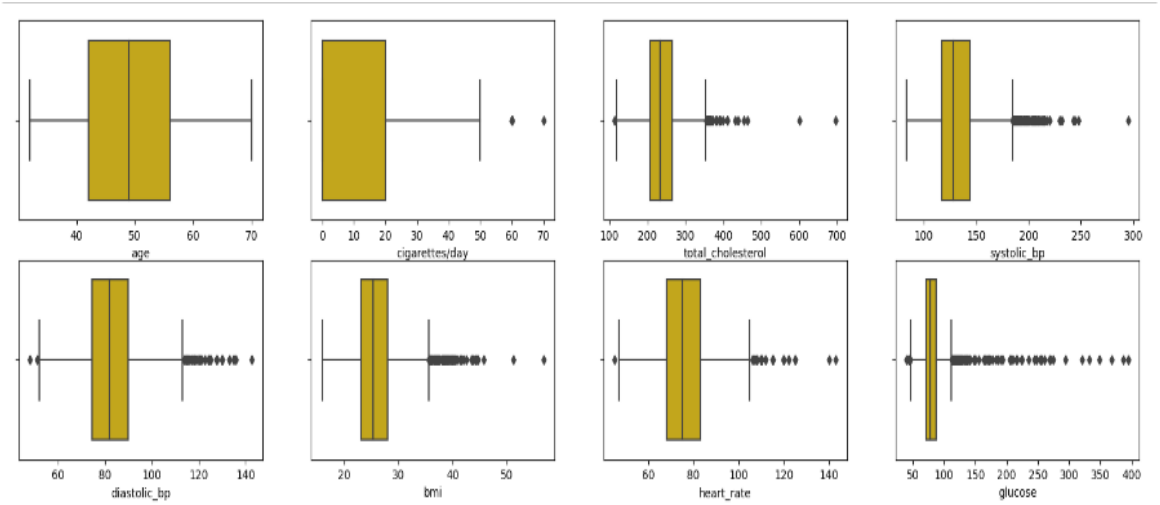
Univariate Analysis of Numerical Features



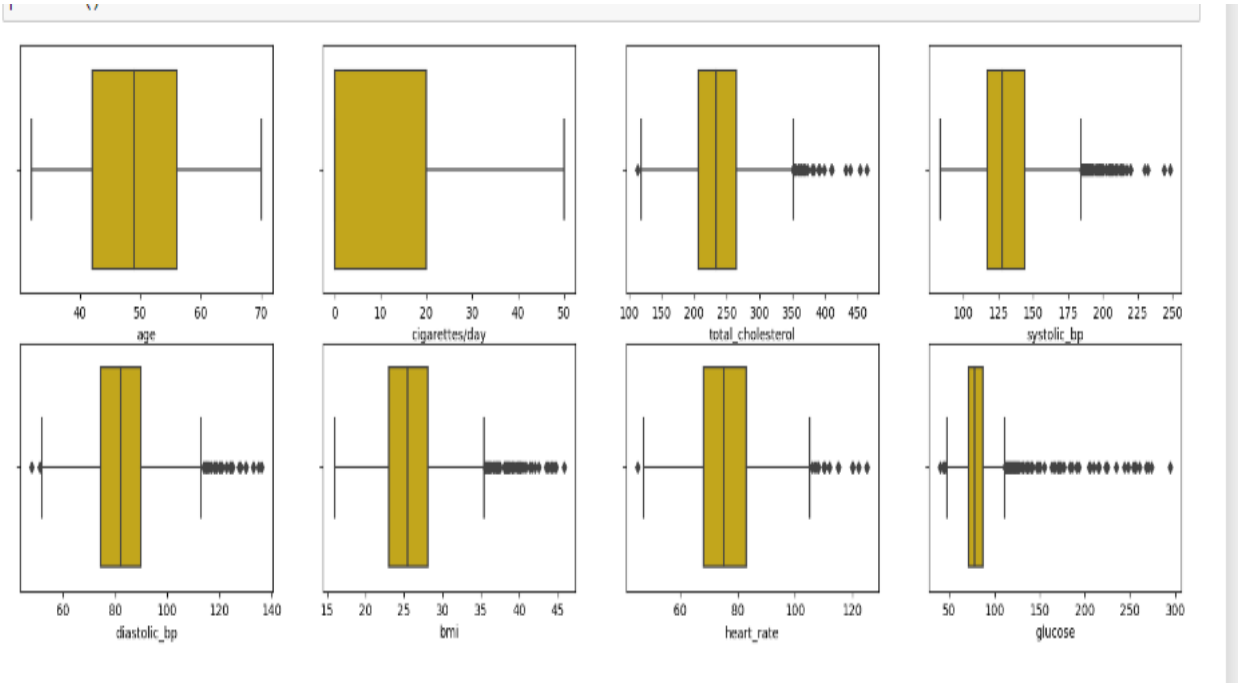
We observed from the above distribution plot of numerical features that the attributes cigarettes/day, total_cholesterol, systolic_bp, bmi, and glucose are slightly right skewed.

Outlier Detection of Numerical Features

Before

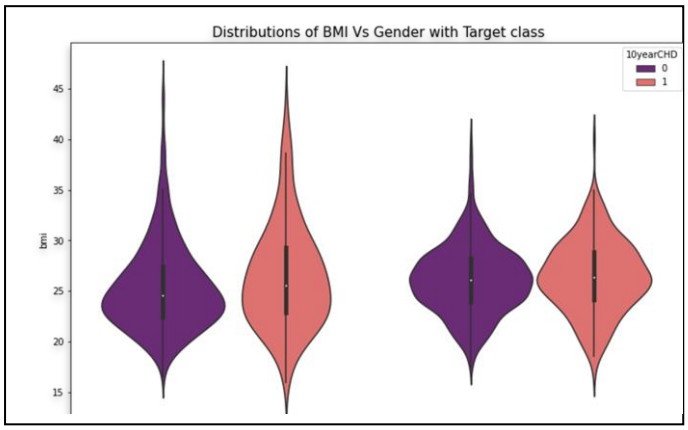


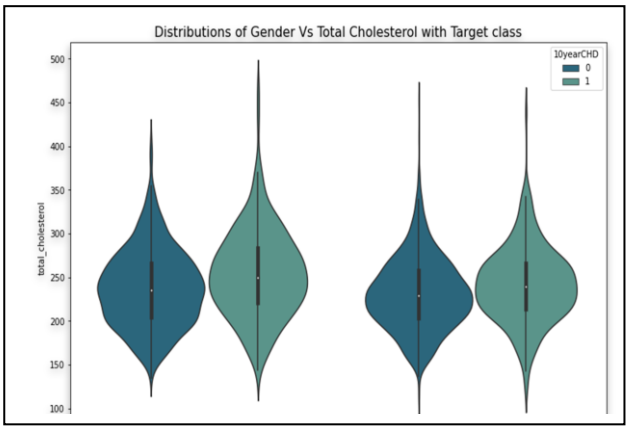
After

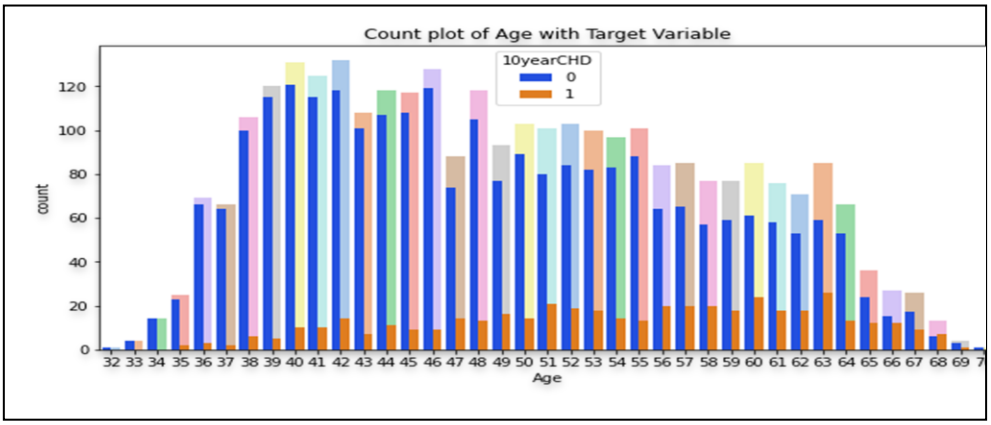


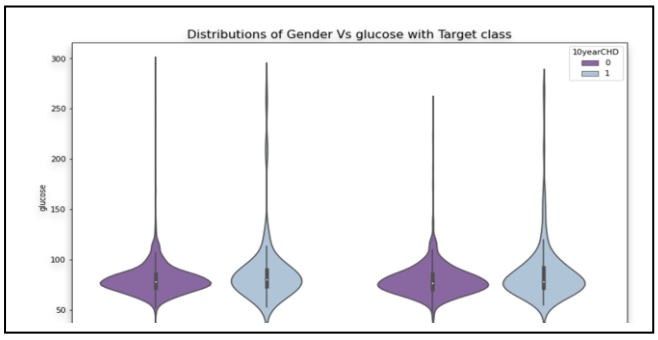
We can see a lot of outliers in columns like, total_cholesterol, systolic_bp, diastolic_bp, bmi, glucose, etc. As stated before we can't manipulate data in such way that we change the original patient stats, neither we can entirely drop those entries with outliers. This will lead to huge amount of data loss, We would lose meaningful data in order to achieve accurate predictions. The best solution to this could only be, to drop the rows with such outliers with minimal data loss.

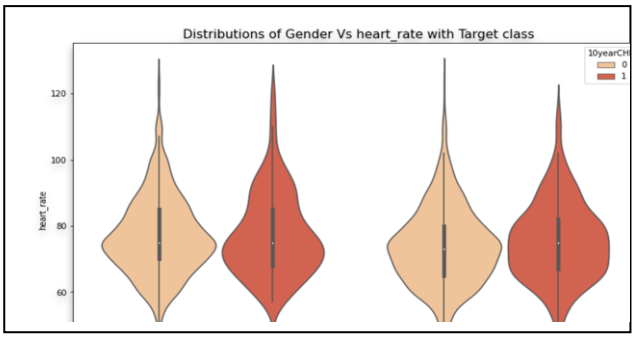
Bivariate Analysis of numerical feature with Target Variable

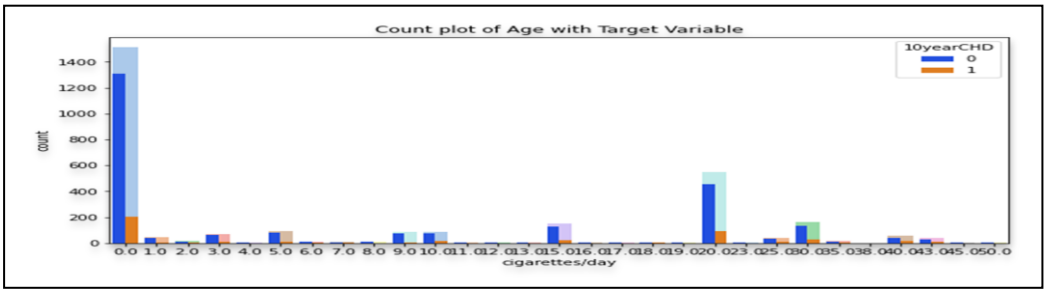




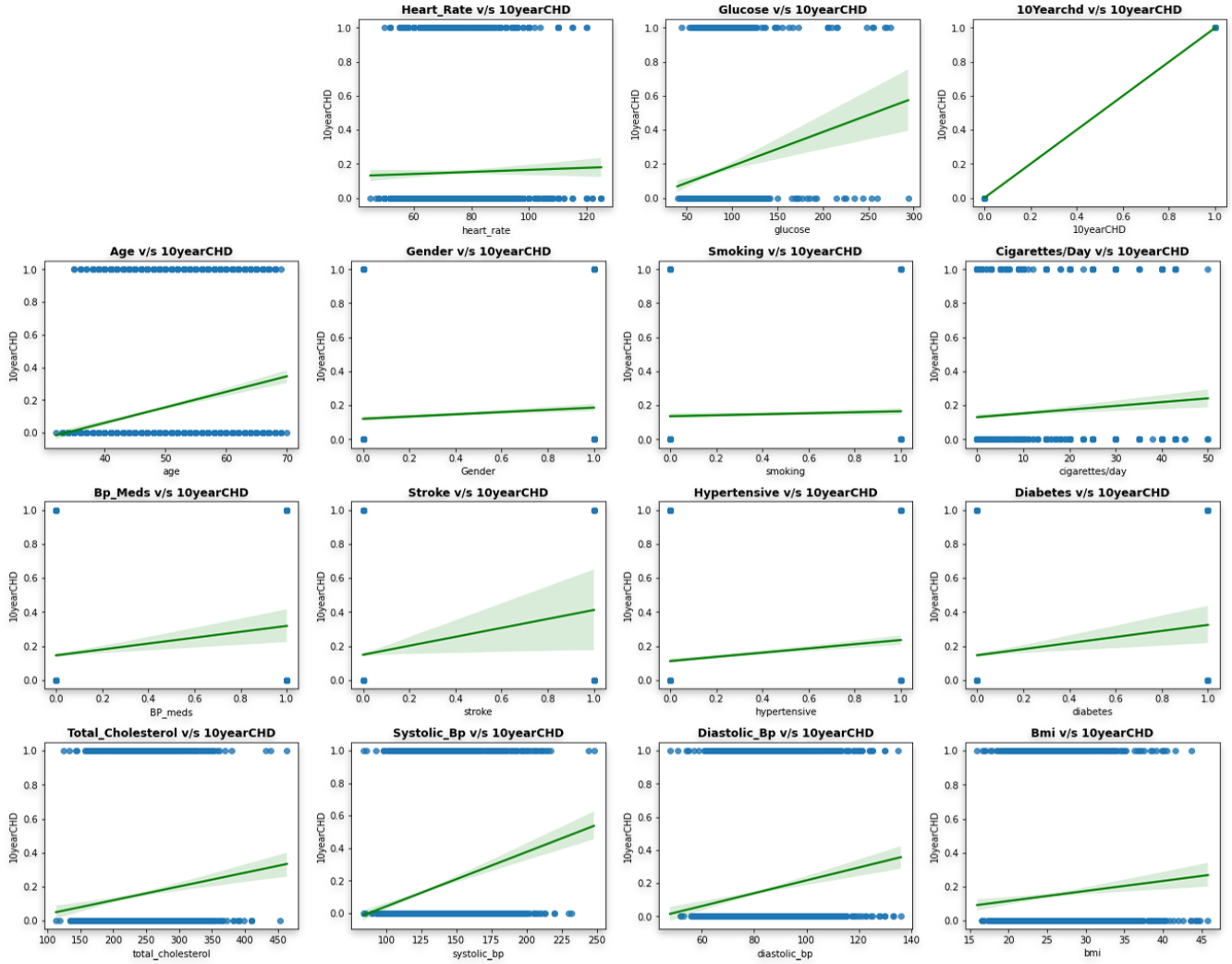






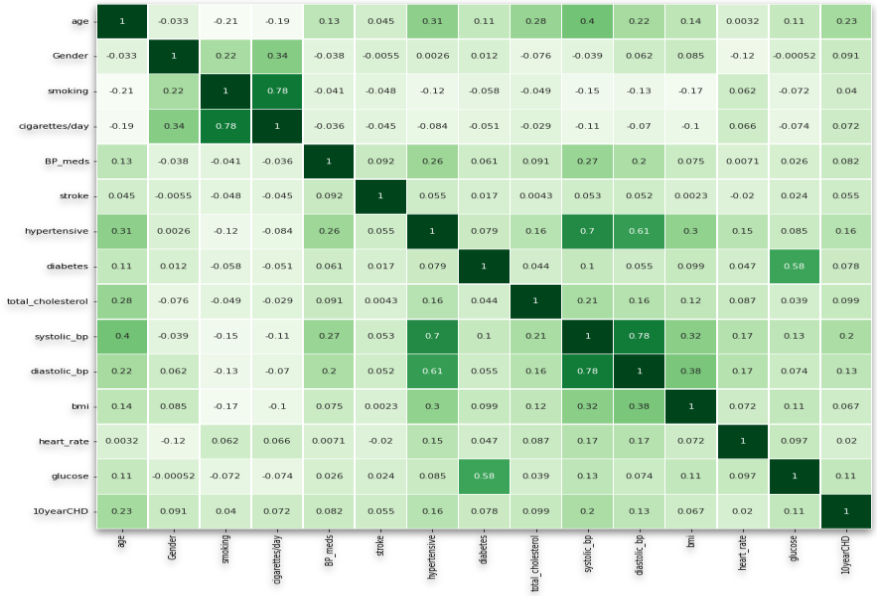


Checking Linearity



Checking Multicollinearity

Before

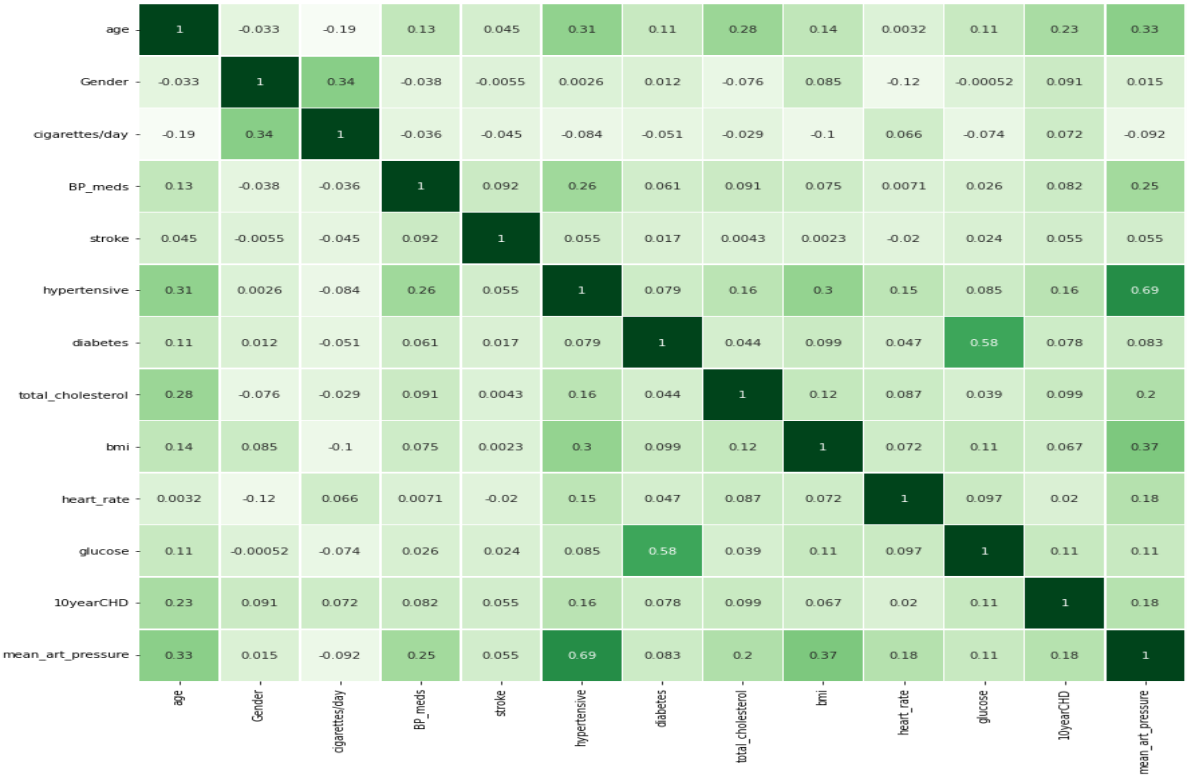


MAP (Mean Arterial Pressure**)= (Systolic Blood Pressure + 2 x Diastolic Blood Pressure) / 3**

```
# Dropping the SysBP and DiaBp attributes, since t
df.drop(columns = ["systolic_bp", "diastolic_bp"],
```

```
#Dropping the smoking.
df.drop(columns = ["smoking"]
```

After



Feature Engineering

All machine learning algorithms use some input data to create outputs. Algorithms require features with some specific characteristics to work properly. Here, the need for feature engineering arises. Feature engineering mainly have two goals:

Preparing the proper input dataset, compatible with the machine learning algorithm requirements.Improving the performance of machine learning models.

We'll try adding and removing some features in this section in order to make a perfect data matrix we can pass to a machine learning model. We will try to interpret categorical features as numeric to be passed to the ML models.

So, These are the code which we have applied for feature engineering

```
#Encoding the 'Gender' feature into binary column.
df['Gender'] = np.where(df['Gender'] == 'M',1,0)

#Encoding the 'smoking' feature into binary column.
df['smoking'] = np.where(df['smoking'] == 'YES',1,0)
```

```
#Creating dummy variables from categorical features.
dataset = pd.get_dummies(df,columns = ['Gender','BP_meds','stroke','hypertensive','diabetes'])
```

```
dataset.shape

(2981, 18)
```

Now, Here is what our dataset looks like after all the transformations.

id	age	cigarettes/day	total_cholesterol	bmi	heart_rate	glucose	10yearCHD	mean_art_pressure	Gender_0	Gender_1	BP_meds_0.0	BP_meds_1.0	stroke_0
1	36	0.0	212.0	29.77	72.0	75.0	0	121.333333	0	1	1	0	1
2	46	10.0	250.0	20.35	88.0	94.0	0	86.000000	1	0	1	0	1
3	50	20.0	233.0	28.26	68.0	94.0	1	111.333333	0	1	1	0	1
4	64	30.0	241.0	26.42	70.0	77.0	0	102.166667	1	0	1	0	1
5	61	0.0	272.0	32.80	85.0	65.0	1	141.333333	1	0	1	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...

stroke_1	hypertensive_0	hypertensive_1	diabetes_0	diabetes_1
0	0	1	1	0
0	1	0	1	0
0	0	1	1	0
0	1	0	1	0
0	0	1	1	0
...	...	...	...	...

Data Preparation

- ❖ Now that the Dataset is cleaned and we have added all the necessary features along with some conversions of categorical features via.,

– Label Encoding

– One Hot Encoding (Dummy Encoding)

- ❖ Then, We used Standard Scaler for transforming data
- ❖ So, now we have split the data into training and testing sets.

– Train Test Split (Test size = “0.25” Random state = “0”)

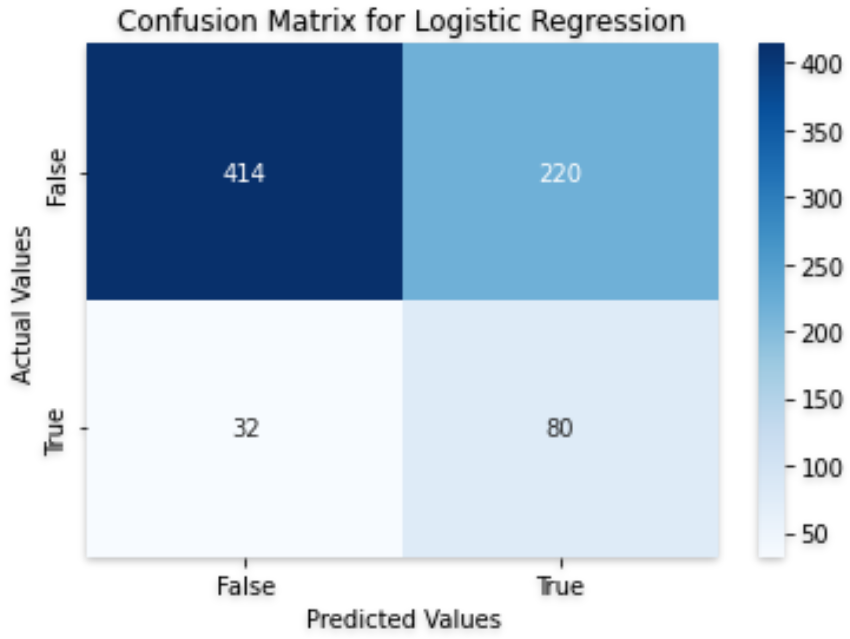
Performance Metrics

- **ROC** also known as Receiver Operating Characteristics, shows the performance of binary class classifiers across the range of all possible thresholds plotting between true positive rate and 1-false positive rate.
- **AUC** measures the likelihood of two given random points, one from positive and one from negative, the classifier will rank the positive points above negative points. AUC-ROC is popular classification metric that presents the advantage of being independent of false positive or negative points.
- **F1 SCORE** is the harmonic mean between Precision and Recall. Macro F1 score is used to know how our model works in overall dataset.
- **Confusion Matrix** gives the count of true negative, true positive, false positive and false negative data points.

Classification Models

- Logistic Regression Classifier
- Decision Tree Classifier
- Random Forest Classifier
- KNN Classifier

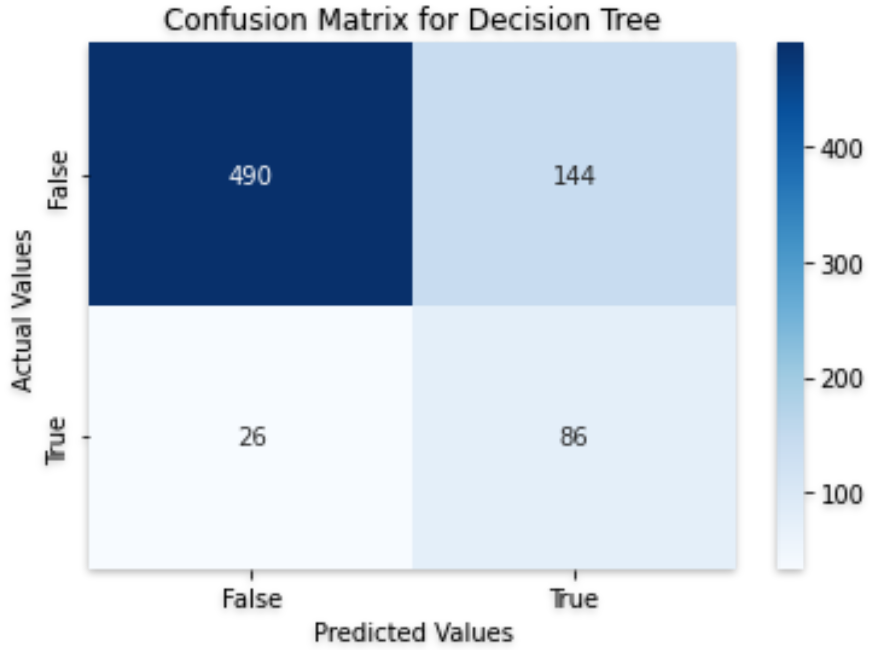
1.Logistic Regression Classifier



Precision	Recall	F1 Score	Support
0.88	1.00	0.93	648

Accuracy on Train data	Accuracy on Test data
0.8535	0.8721

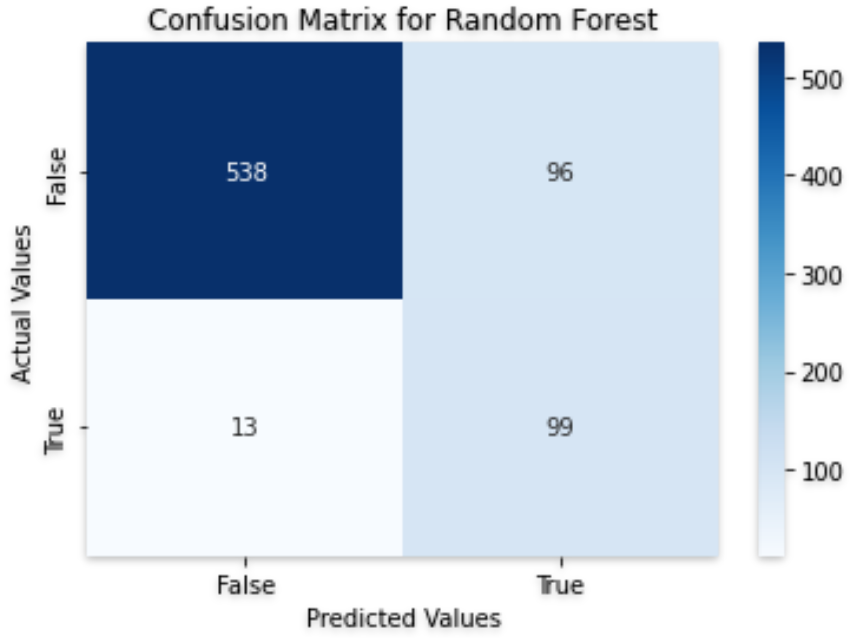
Decision Tree Classifier



Precision	Recall	F1 Score	Support
0.89	0.82	0.85	648

Accuracy on Train data	Accuracy on Test data
1	0.7493

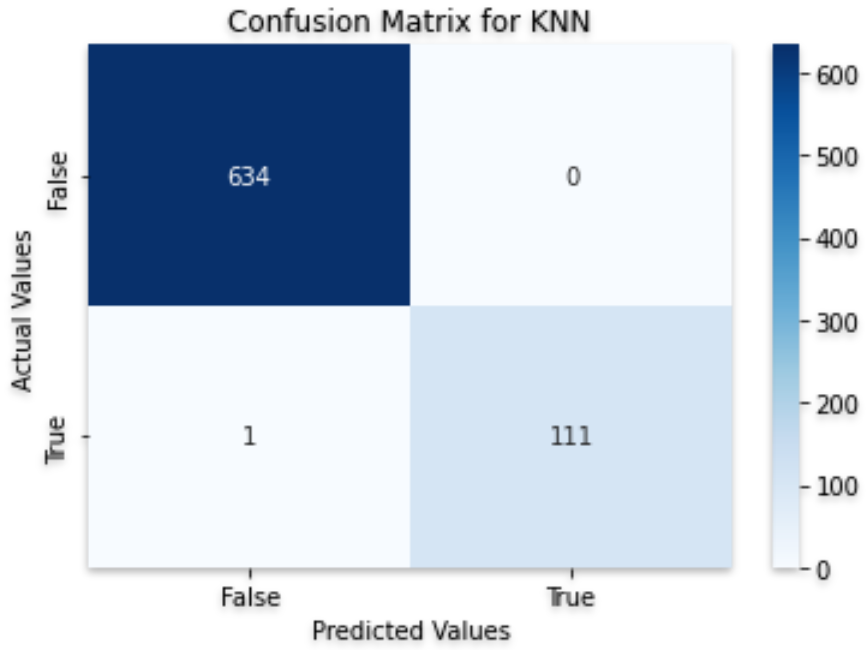
Random Forest Classifier



Precision	Recall	F1 Score	Support
0.88	0.98	0.93	648

Accuracy on Train data	Accuracy on Test data
1	0.8659

K- Nearest Neighbors Classifier



Precision	Recall	F1 Score	Support
0.87	0.97	0.92	648

Accuracy on Train data	Accuracy on Test data
0.8626	0.8552

Analysis of the Result

We used precision, F1-score, recall and accuracy evaluation metrics for evaluating our models. False Positive(FP) is when a model incorrectly predicts a positive outcome. False Negative(FN) is when a model incorrectly predicts the negative outcome. True Positive(TP) is when model correctly predicts a positive outcome. True Negative(TN) is when a model correctly predicts a negative outcome. Precision=TP/(TP+FP) Recall = TP / (TP + FN)

Machine Learning Models	**Accuracy**
Logistic Regression	**87%**
Decision Tree	**74%**
Random Forest	**86%**
KNN	**85%**

Conclusions

- In conclusion, All the features provided in the dataset are extremely important and contribute towards the risk of getting CHDs. Although, we can conclude some majorly important features like:
- As age increases the risk of getting diagnosed with heart disease also increases.
- Cigarette consumption is also a major factor that causes CHDs.
- Patients having Diabetes and cholesterol problems show a higher risk of CHDs.
- Patients having high glucose levels are more prone to CHDs.
- Patients with a history of “strokes” have a higher chance of developing CHDs.
- Patients with high BMI(Body Mass Index) are at more risk of getting diagnosed with CHDs.
- Finally we can say that, Logistic Regression Classifier has performed best among all the models with the **accuracy of 87%** & f1-score of 0.9745.

References

UMAIR SHAFIQUE, FIAZ MAJEED, HASEEB QAISER, IRFAN UL MUSTAFA “DATA MINING IN HEALTHCARE FOR HEART DISEASES”, INTERNATIONAL JOURNAL OF INNOVATION AND APPLIED STUDIES ISSN 2028-9324 VOL. 10, ISSUE 4, (2015), PP. 1312-1322

MALKARI BHARGAV AND J. RAGHUNATH, “A STUDY ON RISK PREDICTION OF CARDIOVASCULAR DISEASE USING MACHINE LEARNING ALGORITHMS”, INTERNATIONAL JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (WWW.JSTOR.ORG), ISSN:2349-5162, VOL.7, ISSUE 8, PAGE NO.683-688, AUGUST 2020

G. SARIKA SINDHU, “ANALYSIS AND PREDICTION OF CARDIOVASCULAR DISEASE USING MACHINE LEARNING CLASSIFIERS”, INTERNATIONAL CONFERENCE ON ADVANCED COMPUTING & COMMUNICATION SYSTEMS (ICACCS) APRIL 2020.