

# PROJECT REPORT

## PREDICTIVE MODELING AND ANALYSIS FOR EFFECTIVE DIAGNOSIS OF CARDIOVASCULAR DISEASES

*Submitted towards the partial fulfillment of the criteria for award of Post Graduate Program  
in Data Analytics by Imarticus*

*Submitted By:*

*S.J. JASON SAMRAJ*

*Course and Batch: Data Analytics &PGA-05*



## **ACKNOWLEDGEMENTS**

We are using this opportunity to express our gratitude to everyone who supported us throughout the project. We are thankful for their inspiring guidance, invaluable constructive criticism, and friendly advice during the project work. We are sincerely grateful to them for sharing their truthful and illuminating views on several issues related to the project.

Further, we were fortunate to have great teachers who readily shared their immense knowledge in data analytics and guided us in a manner that the outcome resulted in enhancing our data skills.

We wish to thank, all the faculties, as this project utilized knowledge gained from the courses formulated by PGA program.

We certify that the work done by us for conceptualizing and completing this project is original and authentic.

## **ABSTRACT**

Cardiovascular diseases are the number 1 cause of death globally, more people die annually from CVDs than from any other cause. An estimated 1.8 crore people died from CVDs in 2016, representing 31% of all global deaths. Of these deaths, 85% are due to heart attack. Over 37% of CVD deaths take place in low- and middle-income countries. Most cardiovascular diseases can be prevented by addressing behavioral risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol using population-wide strategies. People at high cardiovascular risk are due to the presence of risk factors such as hypertension, diabetes, hyperlipidemia or already established disease need early detection and management using counselling and medicines,

**Keywords: Cardiovascular, Predictive modelling, Data visualization, Outlier analysis ,Logistic regression, regression, K-Mean, Navies Bayes ,Decision Tree RandomForest and XGboost.**

## **CERTIFICATE OF COMPLETION**

Certified this Capstone Project<sup>1</sup> titled “Predictive modelling and analysis for effective diagnosis of cardiovascular diseases” is the bona fide work of “S.J. Jason Samraj” from PGA-05 has undertaken and carried out the project under my supervision.

Date: 02-June-2021

Place: Coimbatore

## Contents

<b>ACKNOWLEDGEMENTS .....</b>	<b>2</b>
<b>ABSTRACT.....</b>	<b>3</b>
<b>CERTIFICATEOFCOMPLETION.....</b>	<b>4</b>
<b>CHAPTER1 .....</b>	<b>6</b>
<b>INTRODUCTION.....</b>	<b>6</b>
<b>1.1 PROJECT TITLE.....</b>	<b>6</b>
<b>1.2 OBJECTIVE .....</b>	<b>6</b>
<b>1.3 NEEDFOR STUDY .....</b>	<b>7</b>
<b>1.4 DATA SET DESCRIPTION .....</b>	<b>7</b>
<b>1.5 TOOLS AND TECHNIQUES.....</b>	<b>8</b>
<b>CHAPTER2 .....</b>	<b>9</b>
<b>DATA PREPRATION AND VISUALIZATION.....</b>	<b>9</b>
<b>2.1 DATAINFORMATION .....</b>	<b>9</b>
<b>2.2 HANDLINGMISSINGVALUES .....</b>	<b>10</b>
<b>2.3 CORRELATION .....</b>	<b>11</b>
<b>2.4 OUTLIER .....</b>	<b>11</b>
<b>2.5 DATAVISUALIZATION.....</b>	<b>12</b>
<b>2.6 LABELENCODING .....</b>	<b>17</b>
<b>CHAPTER3 .....</b>	<b>18</b>
<b>BUILDINGMODEL .....</b>	<b>18</b>
<b>TASK1:CVD ANALYSIS [CLEAN DATA].....</b>	<b>18</b>
<b>3.1 X-Y SPLITFORTASK1.....</b>	<b>18</b>
<b>3.2 TRAIN TESTSPLITFOR TASK 1.....</b>	<b>18</b>
<b>3.3 LOGISTICREGRESSIONMODEL.....</b>	<b>19</b>
<b>3.4 DECISION TREE AND RANDOM FOREST CLASSIFER.....</b>	<b>22</b>
<b>3.5 XGBOOSTCLASSIFIER.....</b>	<b>25</b>
<b>3.6 NAVIE BAYES CLASSIFIER .....</b>	<b>26</b>
<b>TASK2:CVD ANALYSIS [OUTLIER DATA] .....</b>	<b>29</b>
<b>3.7 UPPER AND LOWER DATA SPLIT .....</b>	<b>29</b>

# CHAPTER 1

## INTRODUCTION

The most important behavioral risk factors of heart disease and stroke are unhealthy diet, physical inactivity, tobacco use and harmful use of alcohol. The effects of behavioral risk factors may show up in individuals as raised blood pressure, raised blood glucose, raised blood lipids, and overweight and obesity. These “intermediate risks factors” can be measured in primary care facilities and indicate an increased risk of developing a heart attack, stroke, heart failure and other complications.

The primary goal of any health infrastructure is to prevent life threatening health condition at it's infancy by using the existing data to model future health risks of particular health conditions. By observing tracking individual heart pressure, glucose level, bmi, lipids levels and gender. To predicting their risk rate and provide them with thresholds to lead a healthy lifestyle and prevent cardiovascular disease.

### 1.1 TITLE & OBJECTIVE OF THE STUDY

The project titled “PREDICTIVE MODELING AND ANALYSIS FOR EFFECTIVE DIAGNOSIS OF CARDIOVASCULAR DISEASES” is under category “Healthcare”, which inspects the patient’s medical information performed across various hospitals. The project primarily focuses on two tasks: ‘Presence of cardiovascular diseases analysis’, which is a binary classification done by using ML-Supervised classification algorithms and predicting.

#### OBJECTIVE:

- Determine whether the patient has a cardiovascular disease or not (cardiovascular diseases analysis)
- Determine whether the outlier patient as a cardiovascular disease or not(Outlier Analysis)
- Visualizing the factor that influence a patient to have a cardiovascular disease

## 1.2 NEED OF THE STUDY

Due to the recent spike in cardiovascular diseases related death across the world, there is a need to control this crisis which has led to huge expenditure in public health as well as personal wealth among individuals. This can be prevented by informing high risk individuals with preventive measure like alternate healthy lifestyle. This will reduce the strain in the health sector and lead to better allocation of health care funds instead using in preventable cardiovascular diseases. In the project, the possibility of a person having a cardiovascular disease is predicted across cleaned and outlier datasets.

## 1.3 DATASET DESCRIPTION

The data structure: 70,000 rows and 12 columns

The following are the details of the columns present in the dataset

Attribute names	Data Type	Categorical Variable Outcomes
Age (days)	Numerical	
Height (cm)	Numerical	
Weight (kg)	Numerical	
Gender	Categorical	1: male,2: female
Systolic blood pressure (mm of hg)	Numerical	
Diastolic blood pressure (mm of hg)	Numerical	
Cholesterol	Categorical	1: normal, 2: above normal, 3: well above normal
Glucose	Categorical	1: normal, 2: above normal, 3: well above normal
Smoking	Categorical	0: smoker,1: non-smoker
Alcohol intake	Categorical	0: No alcohol,1: alcohol
Physical activity	Categorical	1: active,2: passive
Cardiovascular disease	Categorical	1: present,2: absent

## 1.4 TOOLS AND TECHNIQUES

**Analytics Tool:** Excel,

**Coding Language:** Python (Jupyter Lab)

**Techniques:** Logistic Regression, Decision Tree, Naive Bayes, Random Forest Regressor, XGBoost



## CHAPTER 2

# DATA PREPRATION AND VISUALIZATION

Data preparation is process of cleaning unwanted data and extracting statistically meaningful insights from the dataset. This includes data transformation, data cleaning, feature selection, and visualize data for better understanding. After, importing necessary libraries and using a usable dataset for model building. The process is listed below:

### 2.1 DATA INFORMATION

**Data importing:** The data is in csv. format and is imported using pandas.

```
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
```

**Variable description:**

**Data info** (data.info()) says about variables data type and non-null column count. It describes the data observations, gives statistical details like mean, standard deviation, minimum, maximum, total variables in the respective columns and percentile ranges.

	count	mean	std	min	25%	50%	75%	max
age(years)	70000.0	53.339349	6.759573	29.58	48.39	53.98	58.43	64.97
gender	70000.0	1.349571	0.476838	1.00	1.00	1.00	2.00	2.00
height	70000.0	164.359229	8.210126	55.00	159.00	165.00	170.00	250.00
weight	70000.0	74.205690	14.395757	10.00	65.00	72.00	82.00	200.00
Systolic blood pressure	70000.0	128.817286	154.011419	-150.00	120.00	120.00	140.00	16020.00
Diastolic blood pressure	70000.0	96.630414	188.472530	-70.00	80.00	80.00	90.00	11000.00
cholesterol	70000.0	1.366871	0.680250	1.00	1.00	1.00	2.00	3.00
glucose	70000.0	1.226457	0.572270	1.00	1.00	1.00	1.00	3.00
smoke	70000.0	0.088129	0.283484	0.00	0.00	0.00	0.00	1.00
alcohol	70000.0	0.053771	0.225568	0.00	0.00	0.00	0.00	1.00
Physical active	70000.0	0.803729	0.397179	0.00	1.00	1.00	1.00	1.00
cardiovascular disease	70000.0	0.499700	0.500003	0.00	0.00	0.00	1.00	1.00

## 2.2 HANDLING DATA TYPES, UNIQUE AND MISSING VALUES

The datasets may have missing values for various reasons. They are often encoded as NaN, blanks or simply as zero. Training a model with a lot of missing and misleading values can affect the fit of the model.

Since the model does not consist of any missing values after using `isnull()` function there is no necessity to use `drop` function. If a column consist of more than 80% null values removing the columns is a must and using `data.dropna()` to remove the null value from row along with the entire row.

The datatype should also be considered and changed if necessary to improve the usability of the dataset and reduce the computational power


data.isnull().sum()		data.info()			
age(years)	0	<class 'pandas.core.frame.DataFrame'>			
gender	0	RangeIndex: 70000 entries, 0 to 69999			
height	0	Data columns (total 12 columns):			
weight	0	#	Column	Non-Null Count	Dtype
Systolic blood pressure	0	---	----	-----	----
Diastolic blood pressure	0	0	age(years)	70000 non-null	float64
cholesterol	0	1	gender	70000 non-null	int64
glucose	0	2	height	70000 non-null	int64
smoke	0	3	weight	70000 non-null	float64
alcohol	0	4	Systolic blood pressure	70000 non-null	int64
Physical active	0	5	Diastolic blood pressure	70000 non-null	int64
cardiovascular disease	0	6	cholesterol	70000 non-null	int64
dtype: int64		7	glucose	70000 non-null	int64
		8	smoke	70000 non-null	int64
		9	alcohol	70000 non-null	int64
		10	Physical active	70000 non-null	int64
		11	cardiovascular disease	70000 non-null	int64

### Unique Variables in the Dataset:

The dataset should be checked for unique value to accommodate for categorical value and numerical values to build a model. And should be categorized to as per standards to get the optimum prediction.

The column name should be refined to avoid syntax error when calling it during later programming.

```
data.columns = data.columns.str.replace(' ', '_')
data.columns = data.columns.str.replace('(', '_')
data.columns = data.columns.str.replace(')', '_')
for col in tqdm(data.columns):
    if data[col].dtype == 'int64':
        print(col, ":", data[col].nunique())
    elif data[col].dtype == 'float64':
        print(col, ":", data[col].nunique())
```



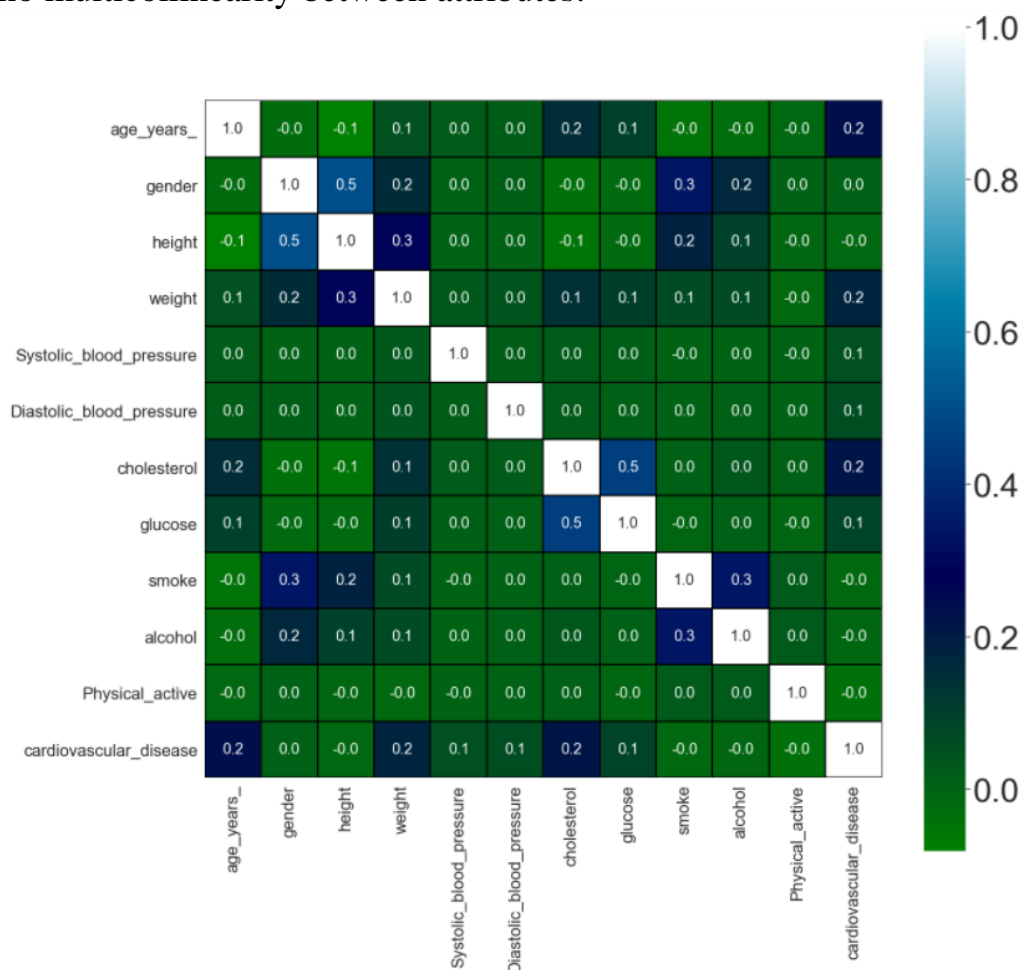
```
age_years_ : 2398
gender : 2
height : 109
weight : 287
Systolic_blood_pressure : 153
Diastolic_blood_pressure : 157
cholesterol : 3
glucose : 3
smoke : 2
alcohol : 2
Physical_active : 2
cardiovascular_disease : 2
```

## 2.3 CORRELATION

The `data.corr()` and heat map function returns the correlation values between each attribute in the dataset in matrix form. Ranges from -1 (highly inverse relationship) to 1 (highly direct relationship) with 0 establishing not relationship. Correlation values plotted as heat map:

### Correlation Inference:

- The highest correlation of 5 can be seen between gender and height and similarly between height and weight, Since physical feature based on gender are well documented this correlation is established.
- A highly linear relationship can be seen between high glucose and cholesterol level
- There is a weak relationship between the dependent variable and the closest relationship which is age, weight and cholesterol
- Since there is no high correlation between attributes, it is safe to assume there is no multicollinearity between attributes.

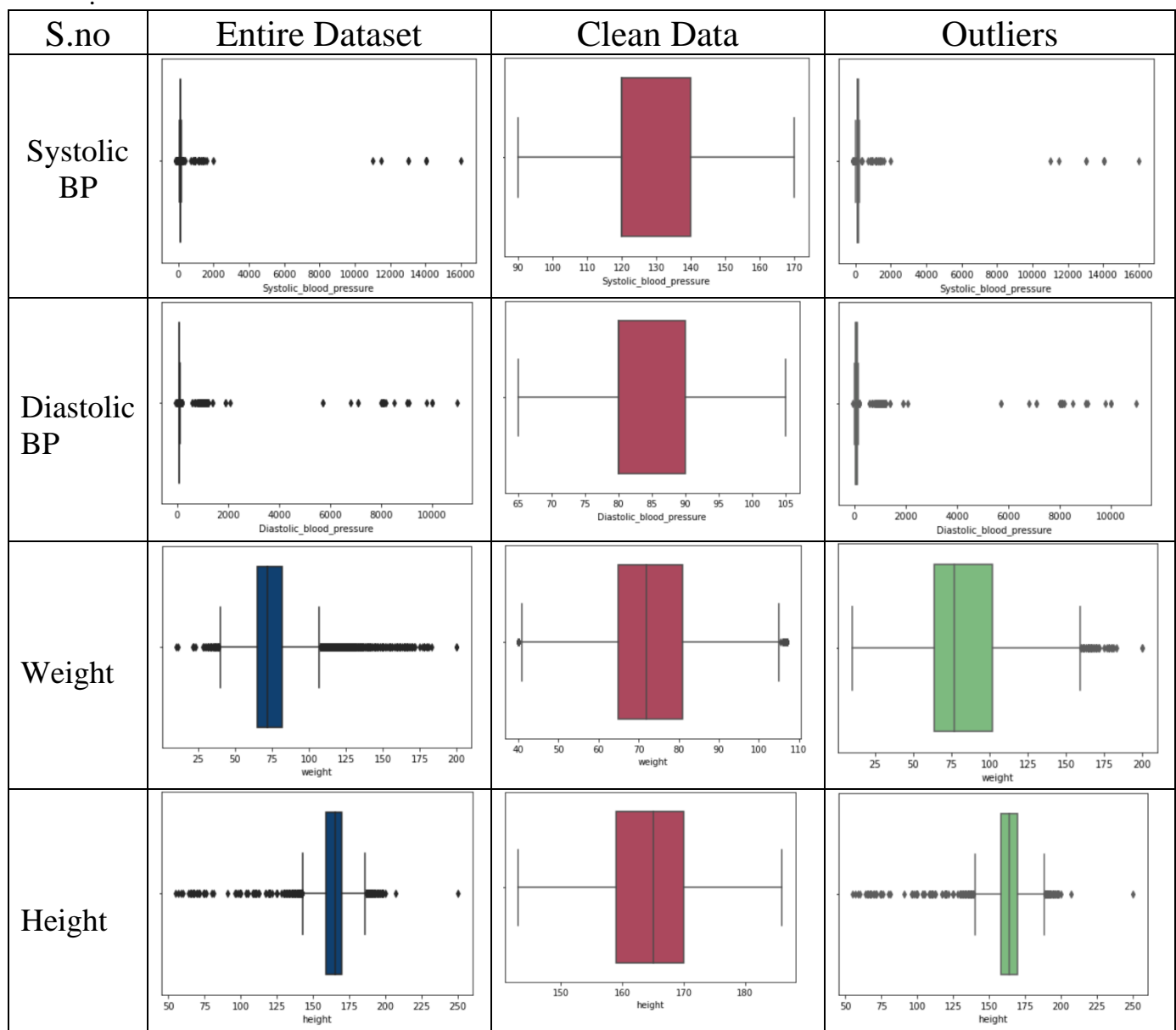


## 2.4 OUTLIERS

Outlier is an observation which don't fall within the normal distribution of a dataset. In statistics, uses Z-score, standard deviation and sigma values to detect outliers. It can be visualized using boxplot as it displays IQR values. General 50 % increase in IQR is consider as the min and max outlier limit. Some of them are not actually outliers, they are also observational values of data which are unique, so a separate outlier analysis can be done for unique values.

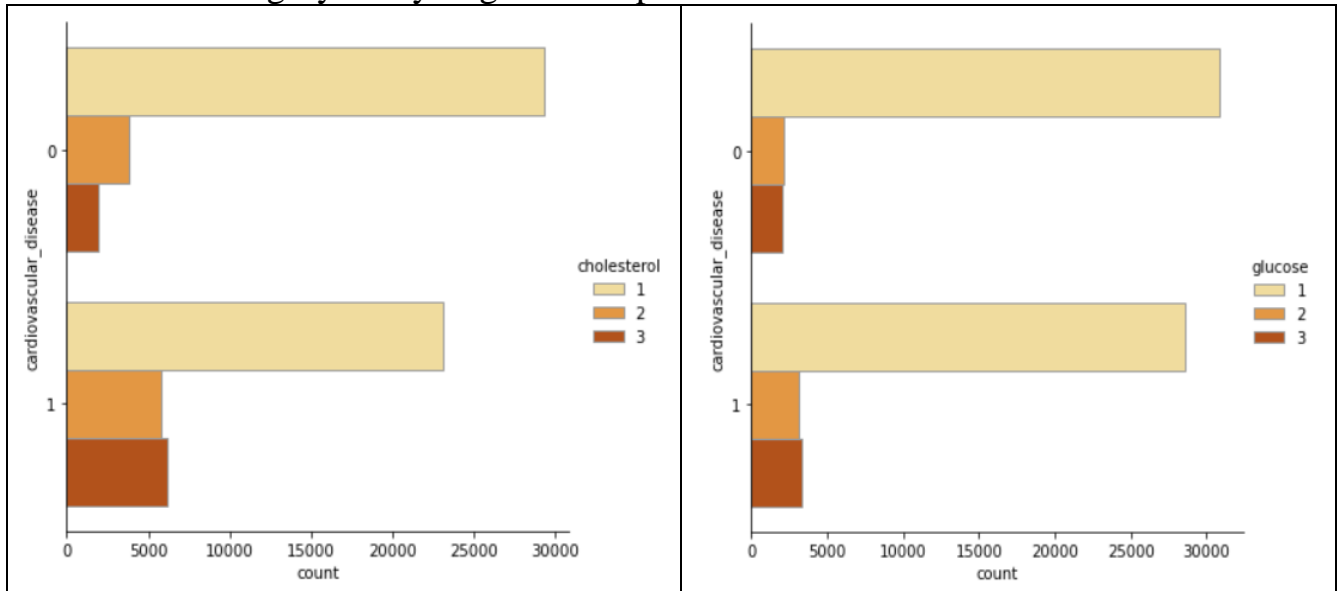
The boxplot of column attributes:

- The numerical value in the dataset provides a good outlier removal detection method

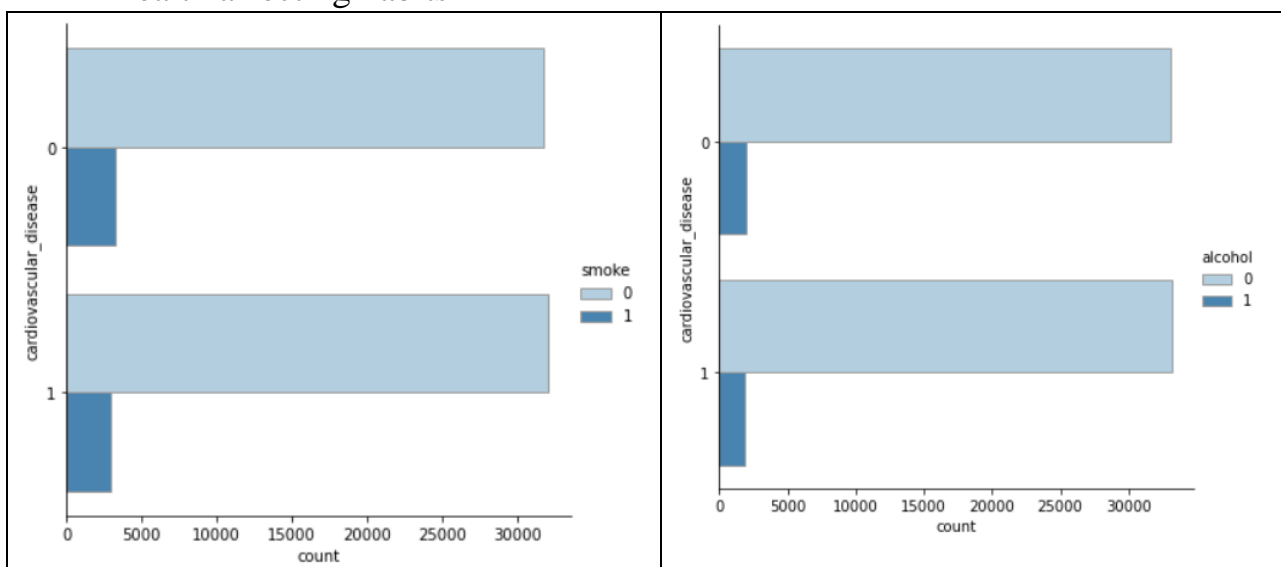


## 2.5 DATA VISUALIZATION

- The count plot provide insight into effect of high glucose and cholesterol level. The number of people with CVD have a comparatively higher percentage population of severe (2) and high risk (3) conditions, with cholesterol having twice as much population when compared to glucose levels.
- All though there is a significant portion of people without CVD are under high-risk category. They might develop future CVD condition

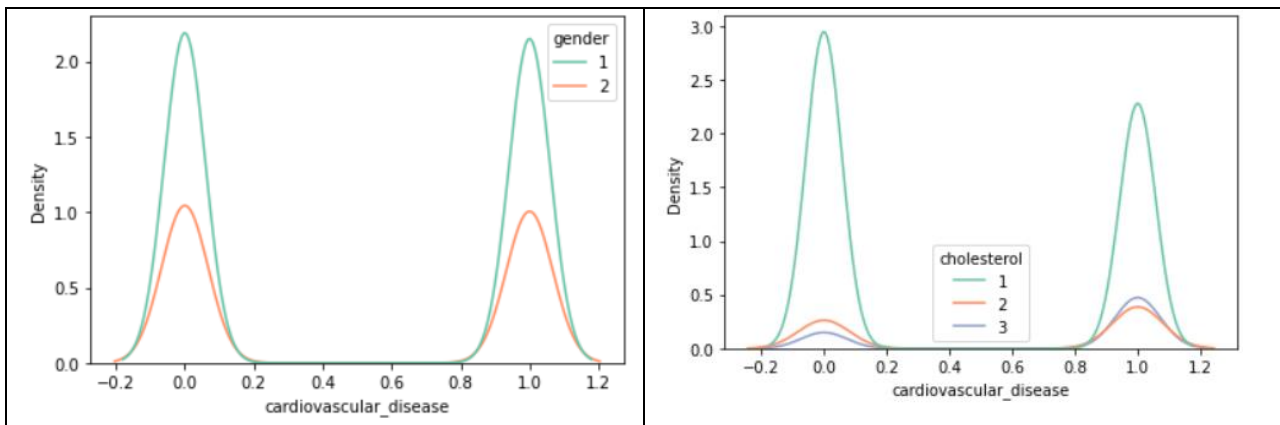


- The percentage of people with and without CVD who have health effecting like alcohol consumption and tobacco consumption seems to be equal. But the frequency of consumption is unknown, so it is inconclusive to state that there is no relationship between CVD's and health affecting habits

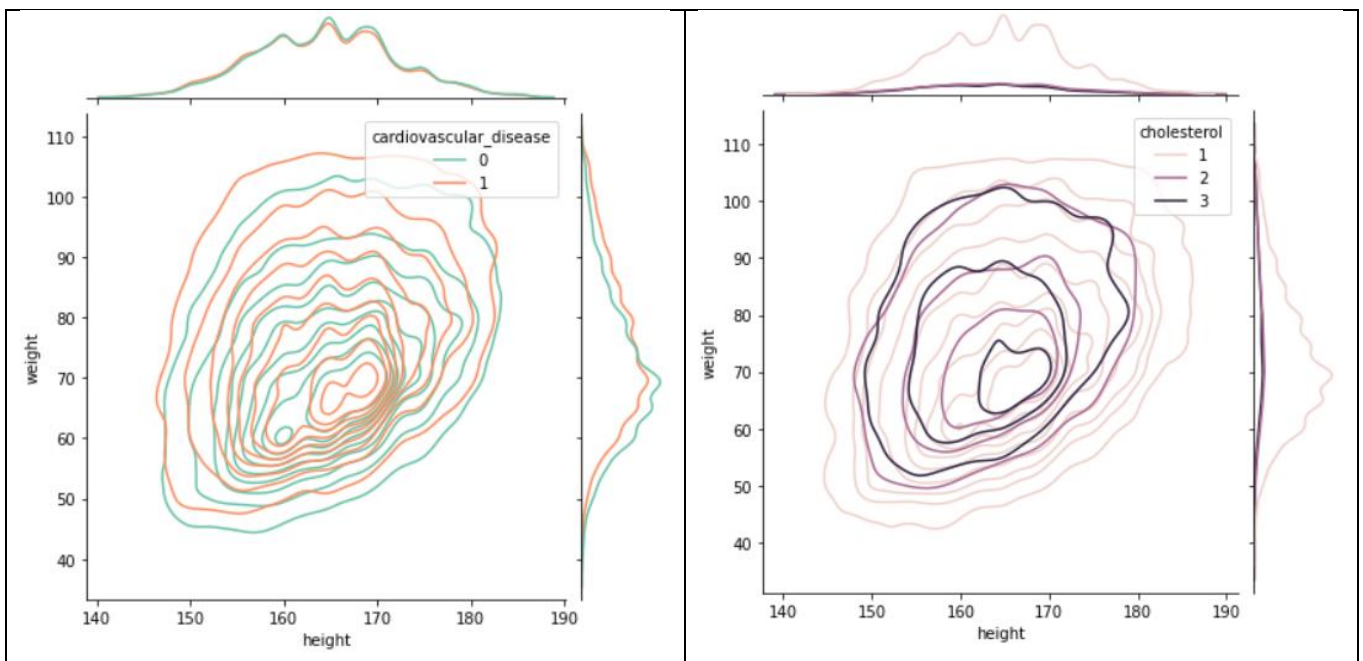


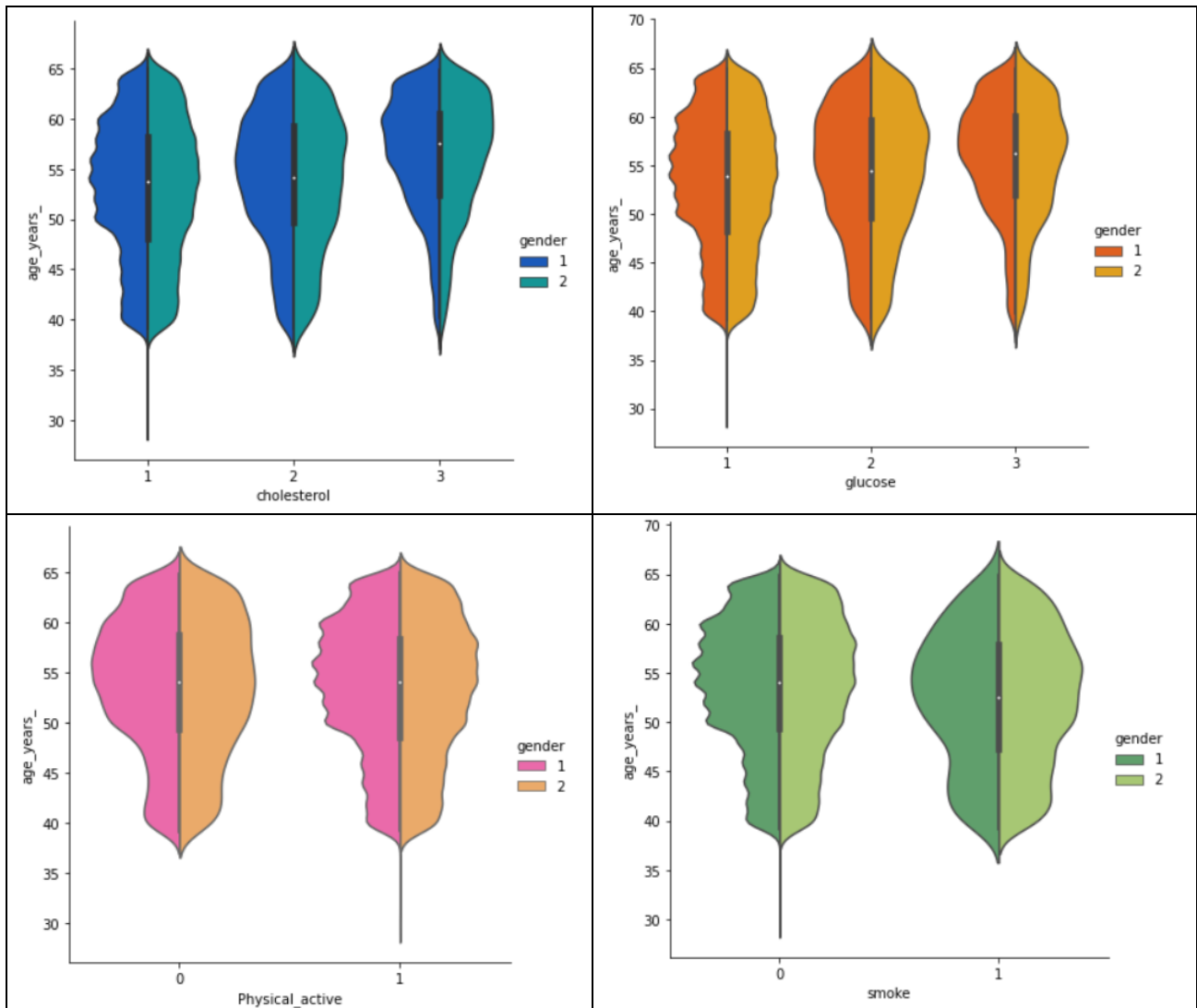
- The density of people with CVD or without based on is gender is similar when similar irrespective of population.
- Based on the density of graph the percentage of people affected by CVD the

portion of severe and high-risk condition curve area are higher

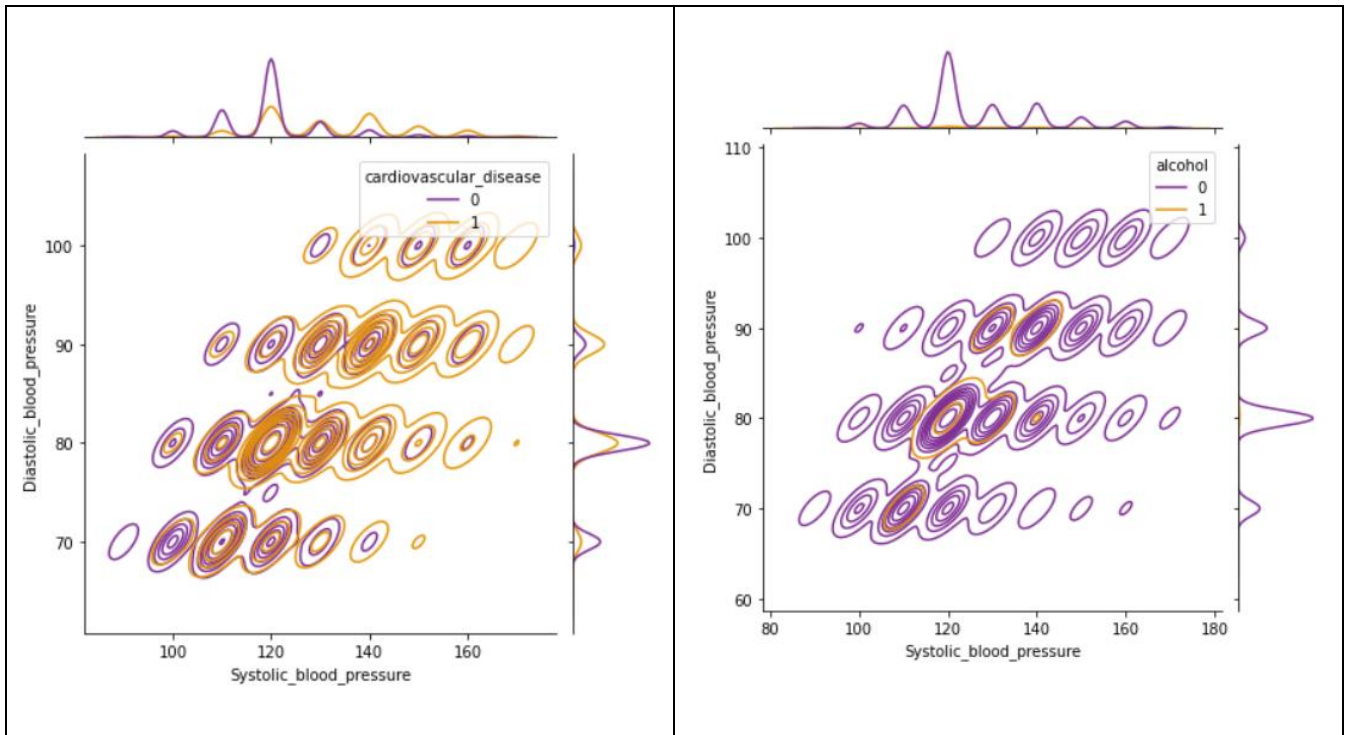


- The CVD distribution across the weight and height of an individual is almost uniform across the height with a slight asymmetry towards the overweight segment in the weight distribution.
- Although people with CVD are uniformly distributed across the weight and height the high-risk cholesterol category is concentrated within a narrow range of weight and height
- The concentration of high-risk CVD people lies within a narrow range, this band is considered normal in body mass index (BMI study). The proportion of CVD lie between normal and obese.

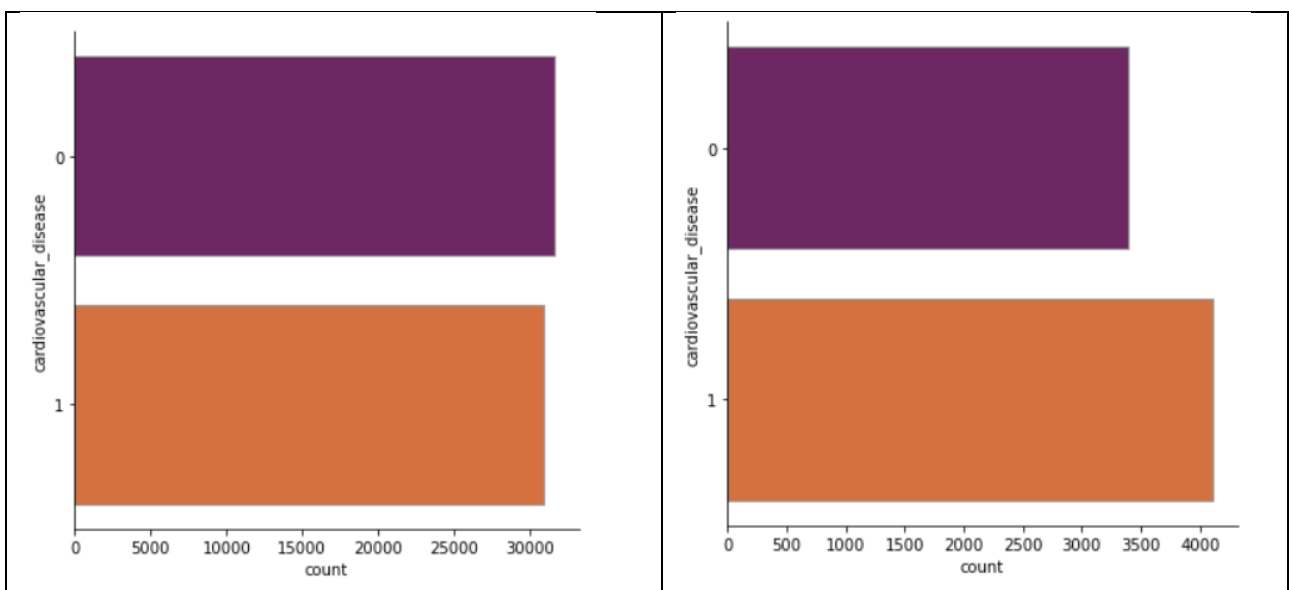




- The graphical similarity of age distribution between the low glucose, cholesterol, active physical life and non-smoker, can tell us that there is a particular segment of across all age group who are involved in health lifestyle which benefits by lowering the CVD risk.
- Similarly, the graphical similarities between severe and high-risk glucose and cholesterol level across the ages are same, which leads us to believes a particular segment of people follow an unhealthy lifestyle with limited physical activities and tobacco usage.



- The favored blood pressure during systolic is 120 and diastolic is 80 which is consider healthy, although majority of the individual's pressure lies between normal it has one of the highest concertation of CVD and the risks increases proportionally as the pressure increases. With higher number of CVD cases occurring at the far end of the graph
- The proportion of people consuming alcohol is very less around high blood pressure zone and the consumption is high within the optimum range of 120 and 80.



- The data that is cleaned has 32,000 case of CVD diseases and the outliers consist of an additional 4000 cases of CVD which constitutes 12.5 % of all cases. Hence an outlier analysis is necessary to estimate the CVD cases which consist of unique variables



## 2.6 LABEL ENCODING

### Blood Pressure Categories



BLOOD PRESSURE CATEGORY	SYSTOLIC mm Hg (upper number)		DIASTOLIC mm Hg (lower number)
NORMAL	LESS THAN 120	and	LESS THAN 80
ELEVATED	120 – 129	and	LESS THAN 80
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 1	130 – 139	or	80 – 89
HIGH BLOOD PRESSURE (HYPERTENSION) STAGE 2	140 OR HIGHER	or	90 OR HIGHER
HYPERTENSIVE CRISIS (consult your doctor immediately)	HIGHER THAN 180	and/or	HIGHER THAN 120

## CONVERTING NUMERICAL TO CATEGORICAL DATA

- Since blood pressure is in numerical format it is convert to a categorical data type to be used in logistics and random forest methods. This is done by replace numerical value with categories of blood pressure presented by American heart association
- Similarly, the numerical value of height and weight is converted into Body Mass Index(bmi) and classified as per standards

```
y=[]
for x in data.Systolic_blood_pressure:
    if x <= 120:
        y.append(1)
    elif x> 120 and x<=130:
        y.append(2)
    elif x> 130 and x<=140:
        y.append(3)
    elif x> 140 and x<=180:
        y.append(4)
    elif x> 180:
        y.append(5)
df=pd.DataFrame({'Systolic_blood_pressure_cat':y})
data['Systolic_blood_pressure_cat']=df
```

```
y=[]
for x in data.Diastolic_blood_pressure:
    if x <= 80:
        y.append(1)
    elif x> 80 and x<=90:
        y.append(2)
    elif x> 90 and x<=120:
        y.append(3)
    elif x> 120:
        y.append(4)
df=pd.DataFrame({'Diastolic_blood_pressure_cat':y})
data['Diastolic_blood_pressure_cat']=df
```

```
y=[]
for x in data.bmi:
    if x <= 18.5:
        y.append(1)
    elif x> 18.5 and x<=25:
        y.append(2)
    elif x> 25 and x<=30:
        y.append(3)
    elif x> 30:
        y.append(4)
df=pd.DataFrame({'bmi_cat':y})
data['bmi_cat']=df
```

## Label Encoder

- Label encoding is the process of converting strings, object type, categorical type variables to numerical form. This process decodes the labels and assigns numerical value, which is understandable by machine
- This done using Sklearn. pre-processing(LabelEncoder)

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
data['gender']=le.fit_transform(data.gender)
data['cholesterol']=le.fit_transform(data.cholesterol)
data['glucose']=le.fit_transform(data.glucose)
data['smoke']=le.fit_transform(data.smoke)
data['alcohol']=le.fit_transform(data.alcohol)
data['Physical_active']=le.fit_transform(data.Physical_active)
data['cardiovascular_disease']=le.fit_transform(data.cardiovascular_disease)
data['Systolic_blood_pressure_cat']=le.fit_transform(data.Systolic_blood_pressure_cat)
data['Diastolic_blood_pressure_cat']=le.fit_transform(data.Diastolic_blood_pressure_cat)
data['bmi_cat']=le.fit_transform(data.bmi_cat)
```

## CHAPTER 3: BUILDING MODEL

### TASK1:CVD ANALYSIS (CLEAN DATA)

The selection of independent variables and dependent variables and splitting train/test data from cleaned dataset with independent variables and dependent variables with a few supervised models which has known label.

#### 3.1 X AND Y SPLIT

The dependent variable(y\_dep) 'cardiovascular disease' and independent variables(x\_ind) are split from dataset, which then used for model fitting. A drop list is used to drop all the numerical variable keeping the categorical form of drop list's variable.

```
y_dep=data_clean.cardiovascular_disease
drop_list=["cardiovascular_disease","age_years_","height","weight","Systolic_blood_pressure","Diastolic_blood_pressure","bmi"]
x_ind=data_clean.drop(drop_list,axis=1)
```

#### 3.2 TRAIN TEST SPLIT

Splitting the dataset randomly to train and test imported from sklearn.model\_selection based on dependent and independent variables. The train test data aids in model performance, trains model on training set and predict seen values(test data)

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,test_size=0.2,random_state=3)
```

80% of data is used as training set and 20% data as test data split upon random state of 2.

```
print("x_test:",x_test.shape,"y_test:",y_test.shape)
```

```
x_test: (1642, 9) y_test: (1642,)
```

```
print("x_train:",x_train.shape,"y_train:",y_train.shape)
```

```
x_train: (6566, 9) y_train: (6566,)
```

### 3.3 LOGISTIC REGRESSION MODEL

Logistic regression is the best binary classification model i.e. used when dependent variable is binary. It is used for finding relationship between one or more continuous/categorical independent variable and one categorical dependent variable.

Logistic regression is the best binary classification model i.e. used when dependent variable is binary. It is used for finding relationship between one or more continuous/categorical independent variable and one categorical dependent variable

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,test_size=0.2,random_state=3)
import statsmodels.api as stm
model_stat=stm.Logit(y_train,x_train).fit()
```

Optimization terminated successfully.  
Current function value: 0.583565  
Iterations 6

From statsmodel.api - logit model is imported, and model summary is obtained. From the summary table significant value(p-value) for independent variables higher than 0.05 is checked.

```
model_stat.summary2()
```

Model:	Logit	Pseudo R-squared:	0.158
Dependent Variable:	cardiovascular_disease	AIC:	57712.7138
Date:	2021-06-01 19:53	BIC:	57791.9892
No. Observations:	49433	Log-Likelihood:	-28847.
Df Model:	8	LL-Null:	-34257.
Df Residuals:	49424	LLR p-value:	0.0000
Converged:	1.0000	Scale:	1.0000
No. Iterations:	6.0000		

	Coef.	Std.Err.	z	P> z	[0.025	0.975]
gender	-0.1176	0.0218	-5.4061	0.0000	-0.1602	-0.0750
cholesterol	0.5091	0.0182	27.9801	0.0000	0.4735	0.5448
glucose	-0.1079	0.0207	-5.2259	0.0000	-0.1484	-0.0675
smoke	-0.2366	0.0403	-5.8676	0.0000	-0.3156	-0.1576
alcohol	-0.2225	0.0492	-4.5240	0.0000	-0.3188	-0.1261
Physical_active	-0.6529	0.0204	-32.0746	0.0000	-0.6928	-0.6130
Systolic_blood_pressure_cat	0.8514	0.0149	57.0069	0.0000	0.8221	0.8806
Diastolic_blood_pressure_cat	0.1388	0.0264	5.2595	0.0000	0.0871	0.1905
bmi_cat	-0.0857	0.0098	-8.7807	0.0000	-0.1049	-0.0666

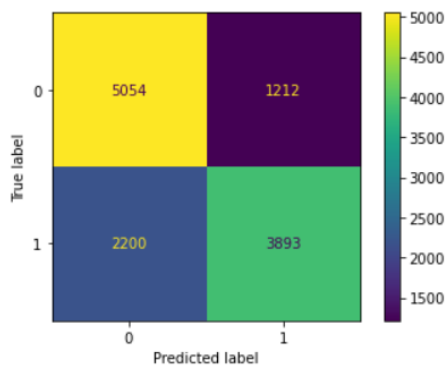
The removal of high p-value attributes is done by focusing on the akaikie information criteria(AIC). Individual attributes with high p-value and a 2% drop in reduction in AIC should be dropped. Then the model is fitted with train dataset.

```
from sklearn.linear_model import LogisticRegression
model_train=LogisticRegression()
model_train.fit(x_train,y_train)
y_pred=model_train.predict(x_test)
y_pred
```

```
array([1, 0, 1, ..., 0, 0, 0], dtype=int64)
```

The performance metrics confusion matrix reveals true positive rate, false positive rate

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(model_train,x_test,y_test)
plt.show()
```



```
model_train.score(x_test,y_test)
```

```
0.7239258839711951
```

Classification report is imported from sklearn.metrics which interprets the accuracy,precision,f1 score results.

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.70	0.81	0.75	6266
1	0.76	0.64	0.70	6093
accuracy			0.72	12359
macro avg	0.73	0.72	0.72	12359
weighted avg	0.73	0.72	0.72	12359

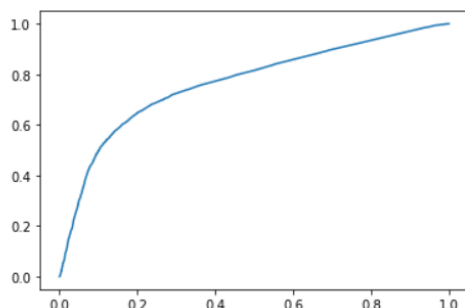
- Precision score means out of total positive predictions (Revenue Positive Sessions), how many are correctly predicted.
- Recall scores mean out of actually revenue positive sessions, how many are correctly predicted
- f1 score represents harmonic mean of precision and recall and takes into account both the values
- ROC-AUCmodel

ROC-AUC receiver operating curve—are under curve is performance

graph, which shows the various threshold values for classification models.

- ROC-AUCcurve:

```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc=roc_auc_score(y_test,model_train.predict(x_test))
fpr,tpr,thresholds=roc_curve(y_test,model_train.predict_proba(x_test)[:,1])
plt.plot(fpr,tpr,logit_roc_auc);
```



```
from sklearn.metrics import accuracy_score
z=LogisticRegression(class_weight="balanced")
z.fit(x_train,y_train)
THRESHOLD=.49
pred=np.where(z.predict_proba(x_test)[:,1]>THRESHOLD,1,0)
pd.DataFrame(data=[accuracy_score(y_test,pred)],index=["accuracy"])
```

```
0
accuracy 0.724411
```

From the graph threshold value of 0.49 chosen and model is trained with set threshold value. Only slight variation in accuracy score is resulted.

### 3.4 DECISION TREE AND RANDOM FOREST CLASSIFIER

#### Decision Tree Classifier:

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,test_size=0.2,random_state=2)
from sklearn import tree
model_DT=tree.DecisionTreeClassifier(random_state=2)
model_DT_fit=model_DT.fit(x_train,y_train)
```

```
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.metrics import confusion_matrix,accuracy_score
acc_list=[]
random_state_list=[]
test_size_list=[]
for g in [.1,.2,.3,.4]:
    for f in range(10):
        x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,test_size=g,random_state=f)
        # fit the model with training set and predict the value
        model_DT=tree.DecisionTreeClassifier(random_state=f)
        model_DT_fit=model_DT.fit(x_train,y_train)
        y_pred=model_DT_fit.predict(x_test)
        acc=accuracy_score(y_test,y_pred)
        #calculate the residual
        acc_list.append(acc)
        random_state_list.append(f)
        test_size_list.append(g)
finallist=pd.DataFrame({'test_size':test_size_list,'random_state':random_state_list,'Accuracy':acc_list})
finallist
```

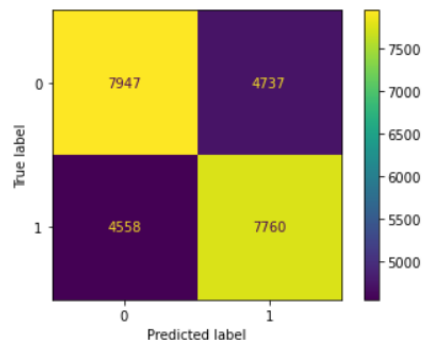
```
DecisionTreeClassifier(max_depth=20, max_features='auto')
```

```
model=tree.DecisionTreeClassifier(criterion='entropy', max_depth=10, max_features='log2',
                                min_samples_split=4)
y_pred=model_DT_fit.predict(x_test)
confusion_matrix(y_test,y_pred)
acc=accuracy_score(y_test,y_pred)
acc
```

0.6282297416206704

## Confusion-matrix

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(model_DT_fit,x_test,y_test)
plt.show()
```



## Classification report

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.71	0.73	0.72	12684
1	0.71	0.69	0.70	12318
accuracy			0.71	25002
macro avg	0.71	0.71	0.71	25002
weighted avg	0.71	0.71	0.71	25002

## Random Forest Classifier:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.

```
from sklearn.ensemble import RandomForestClassifier
model_rf=RandomForestClassifier(random_state=2,n_estimators=200)
model_rf_fit=model_rf.fit(x_train,y_train)
y_pred=model_rf_fit.predict(x_test)
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
confusion_matrix(y_test,y_pred)
```

```
array([[9244, 3440],
       [3851, 8467]], dtype=int64)
```

```
acc=accuracy_score(y_test,y_pred)
acc
```

0.7083833293336533

## Hyper Tuning:

We can hyper tune the model and could try to increase the accuracy of the model. Now we are going to tune the model setting hyper parameter. We set the parameter and fit the parameter into RandomizedSearchCV function where an optimum solution is provide by the algorithm.

```
from sklearn.model_selection import RandomizedSearchCV
parameters={'n_estimators':(100,200,300,400,800),'criterion':('gini','entropy'),'max_features'
rf_grid=RandomizedSearchCV(RandomForestClassifier(),param_distributions=parameters,cv=5)
rf_grid.fit(x_train,y_train)
rf_grid.best_estimator_
```

```
RandomForestClassifier(min_samples_split=6, n_estimators=800)
```

```
best_model=RandomForestClassifier(min_samples_split=6, n_estimators=800)
y_pred=model_rf_fit.predict(x_test)
confusion_matrix(y_test,y_pred)
```

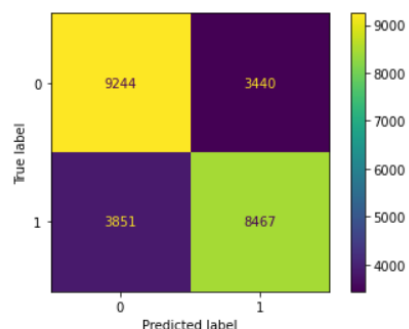
```
array([[9244, 3440],
       [3851, 8467]], dtype=int64)
```

```
acc=accuracy_score(y_test,y_pred)
acc
```

```
0.7083833293336533
```

## Confusion matrix

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(model_rf_fit,x_test,y_test)
plt.show()
```



## Classification report

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.71	0.73	0.72	12684
1	0.71	0.69	0.70	12318
accuracy			0.71	25002
macro avg	0.71	0.71	0.71	25002
weighted avg	0.71	0.71	0.71	25002

### 3.5 XGBOOSTCLASSIFIER

XGBoost is an ensemble learning method. Ensemble learning offers a systematic solution to combine the predictive power of multiple learners. The resultant is a single model which gives the aggregated output from several models.

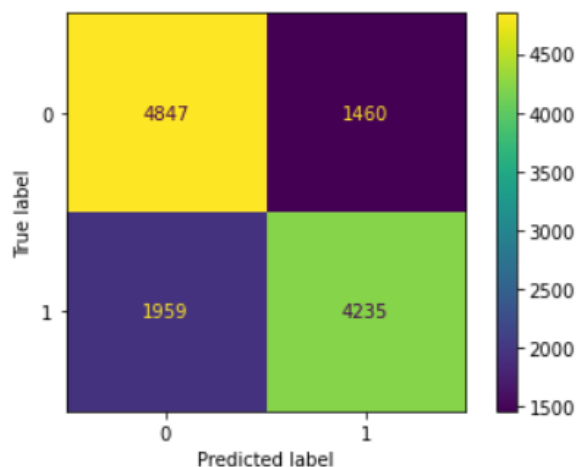
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,test_size=0.2,random_state=2)
model =XGBClassifier()
model.fit(x_train,y_train)
predict_train = model.predict(x_train)
```

```
C:\Users\Jason\anaconda3\lib\site-packages\xgboost\sklearn.py:1146: UserWarning: The use
eprecated and will be removed in a future release. To remove this warning, do the followi
also when constructing XGBClassifier object; and 2) Encode your labels (y) as integers st
_class - 1].
warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
[00:18:17] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.4.0/src/lear
0, the default evaluation metric used with the objective 'binary:logistic' was changed fr
t eval_metric if you'd like to restore the old behavior.
```

```
# Accuracy Score on train dataset
accuracy_train = accuracy_score(y_train,predict_train)
accuracy_train
```

0.7626789856811456



	precision	recall	f1-score	support
0	0.71	0.77	0.74	6307
1	0.74	0.68	0.71	6194
accuracy			0.73	12501
macro avg	0.73	0.73	0.73	12501
weighted avg	0.73	0.73	0.73	12501



### 3.6 NAVIES BAYES CLASSIFIER

Naive bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong independence assumptions between the features. The data is initial Standardize by removing the mean and scaling to unit variance

```
# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x_ind,y_dep,test_size=0.2,random_state=3)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

Then model is fitted in anyone of the bayes method like GaussianNB, BernoulliNB, MultinomialNB, ComplementNB, CategoricalNB

```
# Training the Naive Bayes model on the Training set
from sklearn.naive_bayes import GaussianNB,BernoulliNB,MultinomialNB,ComplementNB,CategoricalNB
classifier = GaussianNB()
classifier.fit(x_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(x_test)

# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score
ac = accuracy_score(y_test,y_pred)
cm = confusion_matrix(y_test, y_pred)
cm
```

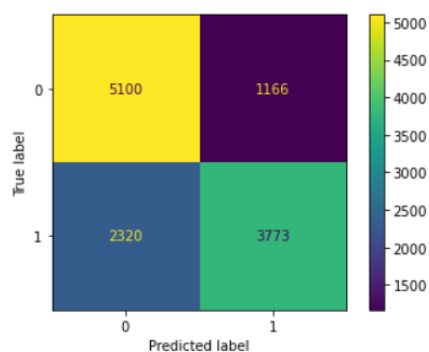
```
array([[5100, 1166],
       [2320, 3773]], dtype=int64)
```

```
classifier.score(x_test,y_test)
```

```
0.7179383445262562
```

Confusion matrix:

```
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(classifier,x_test,y_test)
plt.show()
```



	precision	recall	f1-score	support
0	0.69	0.81	0.75	6266
1	0.76	0.62	0.68	6093
accuracy			0.72	12359
macro avg	0.73	0.72	0.71	12359
weighted avg	0.73	0.72	0.72	12359

## TASK 2: OUTLIER CVN PREDICTION

The outlier which was removed from the initial dataset consisted of 12.5% of all the CVD cases in the dataset, it will a critical oversight to remove them as outlier. They can be considered as unique variables with using information.

Since the outlier are split form both extremes of the dataset it advisable to have two separate analyses. The two set are split into upper and lower segments

### 3.1UPPER AND LOWER DATA SPLIT

```
data_out_lower = data[(data < (Q1 - 1.5 * IQR)).any(axis=1)]
data_out_lower
```

```
data_out_upper = data[(data > (Q3 + 1.5 * IQR)).any(axis=1)]
data_out_upper
```

### RANDOM FOREST CLASSIFER [FOR OUTLIER]

```
array([[ 56, 136],
       [ 35, 612]], dtype=int64)
```

```
acc=accuracy_score(y_test,y_pred)
acc
```

```
0.7961859356376639
```

```
array([[456,  50],
       [110,  73]], dtype=int64)
```

```
acc=accuracy_score(y_test,y_pred)
acc
```

```
0.7677793904208998
```

### NAVIE BAYES CLASSIFER [FOR OUTLIER]

```
array([[429,  49],
       [132,  79]], dtype=int64)
```

```
classifier.score(x_test,y_test)
```

```
0.737300435413643
```

```
array([[116, 147],
       [ 91, 652]], dtype=int64)
```

```
classifier.score(x_test,y_test)
```

```
0.7634194831013916
```

## XGBOOST CLASSIFIER [FOR OUTLIER]

```
accuracy_train = accuracy_score(y_train,predict_train)
accuracy_train
```

0.9341085271317829

```
predict_test = model.predict(x_test)
```

```
accuracy_test = accuracy_score(y_test,predict_test)
accuracy_test
```

0.7783075089392133

```
accuracy_train = accuracy_score(y_train,predict_train)
accuracy_train
```

0.9341085271317829

```
predict_test = model.predict(x_test)
```

```
accuracy_test = accuracy_score(y_test,predict_test)
accuracy_test
```

0.7783075089392133

## **CONCLUSION**

From the cardio-vascular diseases dataset. We have predicted CVD among 70,000 individuals from a cleaned dataset and outlier dataset. This will aid in developing better predictive tool for diagnostics of CVD in future and also establish the relationships between the factor that affect CVD

### **BEST MODELS[CLEANED DATA SET]:**

<b>Model Used</b>	<b>Accuracy</b>
Logistics	72.44%
Decision Tree	62.82%
Random forest	70.83%
XGBoost	73.64%
Navie Bayes	71.93%

For the cleaned dataset used for CVD prediction, XGBoost provided the best accuracy of over 73.64% on the dataset. This model is chosen over Logistics and Navie Bayes model because it improves accuracy of prediction with loss function.

### **BEST MODELS[OUTLIER DATA SET]:**

<b>Model Used</b>	<b>Upper Dataset</b>	<b>Lower Dataset</b>
Random Forest	79.61%	76.77%
Navie Bayes	73.77%	76.34%
XGBoost	77.83%	77.83%

In case of outlier model, Random Forest model is selected, as the model produce highest combined accuracy for both the dataset. It is closely followed by XGBoost combines advantages of both random forest and gradient boosting and provide an accuracy.