HEART DISEASE PREDICTION

A PROJECT REPORT

In partial fulfilment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Under the guidance of

Ms. DEBASREE MITRA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, JIS COLLEGE OF ENGINEERING.

Block-A, Phase-III, Kalyani, Nadia, Pin-741235

West Bengal, India

June, 2023

CERTIFICATE

This is to certify that **MAINAK GHOSH (123211003072),PRITHA GUPTA (123211003099),MAYANK VAIBHAV (123211003075)** have completed their project entitled **HEART DISEASE PREDICTION,** under the guidance of **Ms. DEBASREE MITRA** in partial fulfilment of the requirements for the award of the **Bachelor of Technology in Computer Science and Engineering** from JIS college of Engineering (An Autonomous Institute) is an authentic record of their own work carried out during the academic year 2022-23 and to the best of our knowledge, this work has not been submitted elsewhere as part of the process of obtaining a degree, diploma, fellowship or any other similar title.

---------------------------------	-------------------------------	------------------------------
Signature of the Supervisor	**Signature of the HOD**	**Signature of the Principal**

Place:

Date:

ACKNOWLEDGEMENT

The analysis of the project work wishes to express our gratitude to Ms. Debasree Mitra for allowing the degree attitude and providing effective guidance in development of this project work. Her conscription of the topic and all the helpful hints, she provided, contributed greatly to successful development of this work, without being pedagogic and overbearing influence.

We also express our sincere gratitude to Dr. Bikramjit Sarkar, Head of the Department of Computer Science and Engineering of JIS College of Engineering and all the respected faculty members of Department of CSE for giving the scope of successfully carrying out the project work.

Finally, we take this opportunity to thank to Prof. **(Dr.) Partha Sarkar**, Principal of JIS College of Engineering for giving us the scope of carrying out the project work.

Date:

……………………………………………………………………….

MAINAK GHOSH

B.TECH in Computer Science and Engineering

2thYEAR/4th SEMESTER

Univ Roll--123211003072

……………………………………………………………………….

PRITHA GUPTA

B.TECH in Computer Science and Engineering

2thYEAR/4th SEMESTER

Univ Roll--123211003099

……………………………………………………………………….

MAYANK VAIBHAV

B.TECH in Computer Science and Engineering

2thYEAR/4th SEMESTER

Univ Roll--123211003075

DECLARATION

We hereby declare that the project work being presented in the project proposal entitled **"HEART DISEASE PREDICTION"** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in **JIS COLLEGE OF ENGINEERING, KALYANI, NADIA, WEST BENGAL,** is an authentic work carried out under the guidance of **Ms. DEBASREE MITRA**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date: 2/7/23

Name of the Student: MAINAK GHOSH
PRITHA GUPTA
MAYANK VAIBHAV

CONTENTS

INTRODUCTION

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and has fewer syntactical constructions than other languages.

Python is interpreted: Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to Perl and PHP.

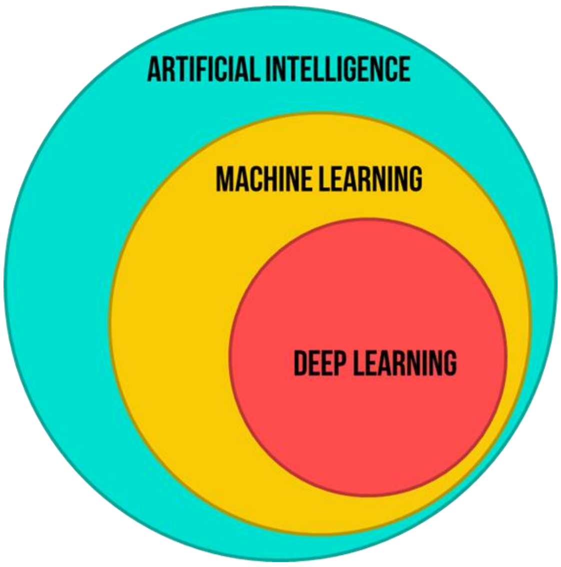
Python is Interactive: You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python is Object-Oriented: Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language: Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

LITERATTURE SURVEY

ARTIFICIAL INTELLIGENCE



According to the father of Artificial Intelligence, John McCarthy, it is *"The science and engineering of making intelligent machines, especially intelligent computer programs".*

Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks, and how humans learn, decide, and work while trying to solve a problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

The development of AI started with the intention of creating similar intelligence in machines that we find and regard high in humans. To Create Expert Systems − The systems which exhibit intelligent behaviour, learn, demonstrate, explain, and advice its users.
To Implement Human Intelligence in Machines − Creating systems that understand, think, learn, and behave like humans.

Applications of AI

AI has been dominant in various fields such as:-

Gaming − AI plays crucial role in strategic games such as chess, poker, tic-tac-toe, etc., where machine can think of large number of possible positions based on heuristic knowledge.

Natural Language Processing − It is possible to interact with the computer that understands natural language spoken by humans.

Expert Systems − There are some applications which integrate machine, software, and special information to impart reasoning and advising. They provide explanation and advice to the users.

Vision Systems − These systems understand, interpret, and comprehend visual input on the computer.

For example: A spying aeroplane takes photographs, which are used to figure out spatial information Or map of the areas. Doctors use clinical expert system to diagnose the patient. Police use computer software that can recognize the face of criminal with the stored portrait made by forensic artist.

Speech Recognition − Some intelligent systems are capable of hearing and comprehending the language in terms of sentences and their meanings while a human talks to it. It can handle different accents, slang words, noise in the background, change in human's noise due to cold, etc.

Handwriting Recognition − The handwriting recognition software reads the text written on paper by a pen or on screen by a stylus. It can recognize the shapes of the letters and convert it into editable text.

Intelligent Robots − Robots are able to perform the tasks given by a human. They have sensors to detect physical data from the real world such as light, heat, temperature, movement, sound, bump, and pressure. They have efficient processors, multiple sensors and huge memory, to exhibit intelligence. In addition, they are capable of learning from their mistakes and they can adapt to the new environment.

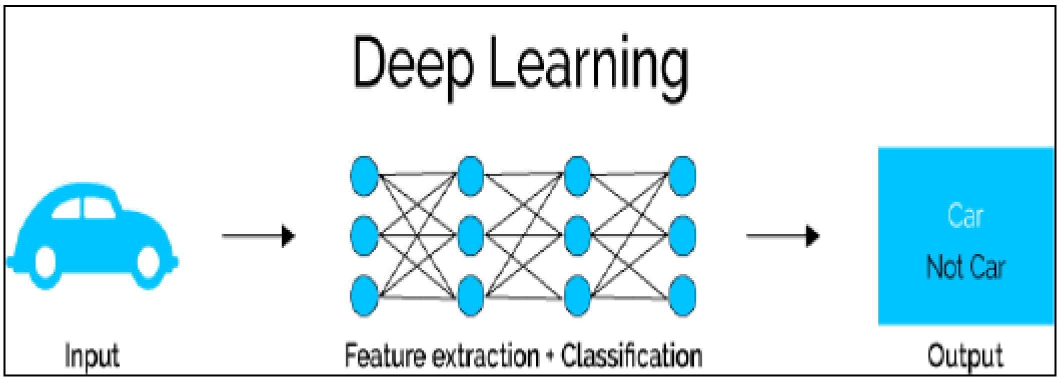
Applications of AI

DEEP LEARNING

Deep learning is a subset of machine learning. Usually, when people use the term deep learning, they are referring to deep artificial neural networks, and somewhat less frequently to deep reinforcement learning.

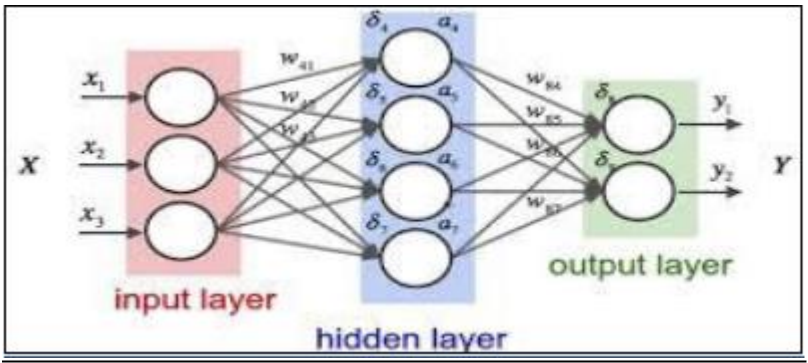
Deep learning is a class of machine learning algorithms that:

- Use a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
- Learn in supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) manners.
- Learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.
- Use some form of gradient descent for training via backpropagation.



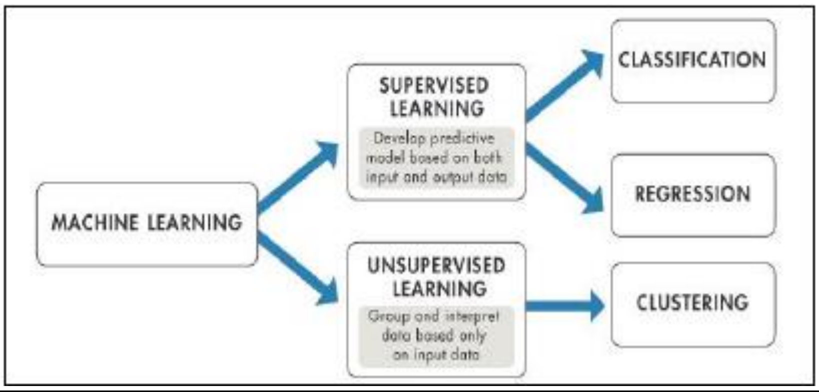
NEURAL NETWORKING

Artificial neural networks (**ANNs**) or **connectionist systems** are computing systems inspired by the biological neural networks that constitute animal brains. Such systems learn (progressively improve performance on) tasks by considering examples, generally without task-specific programming



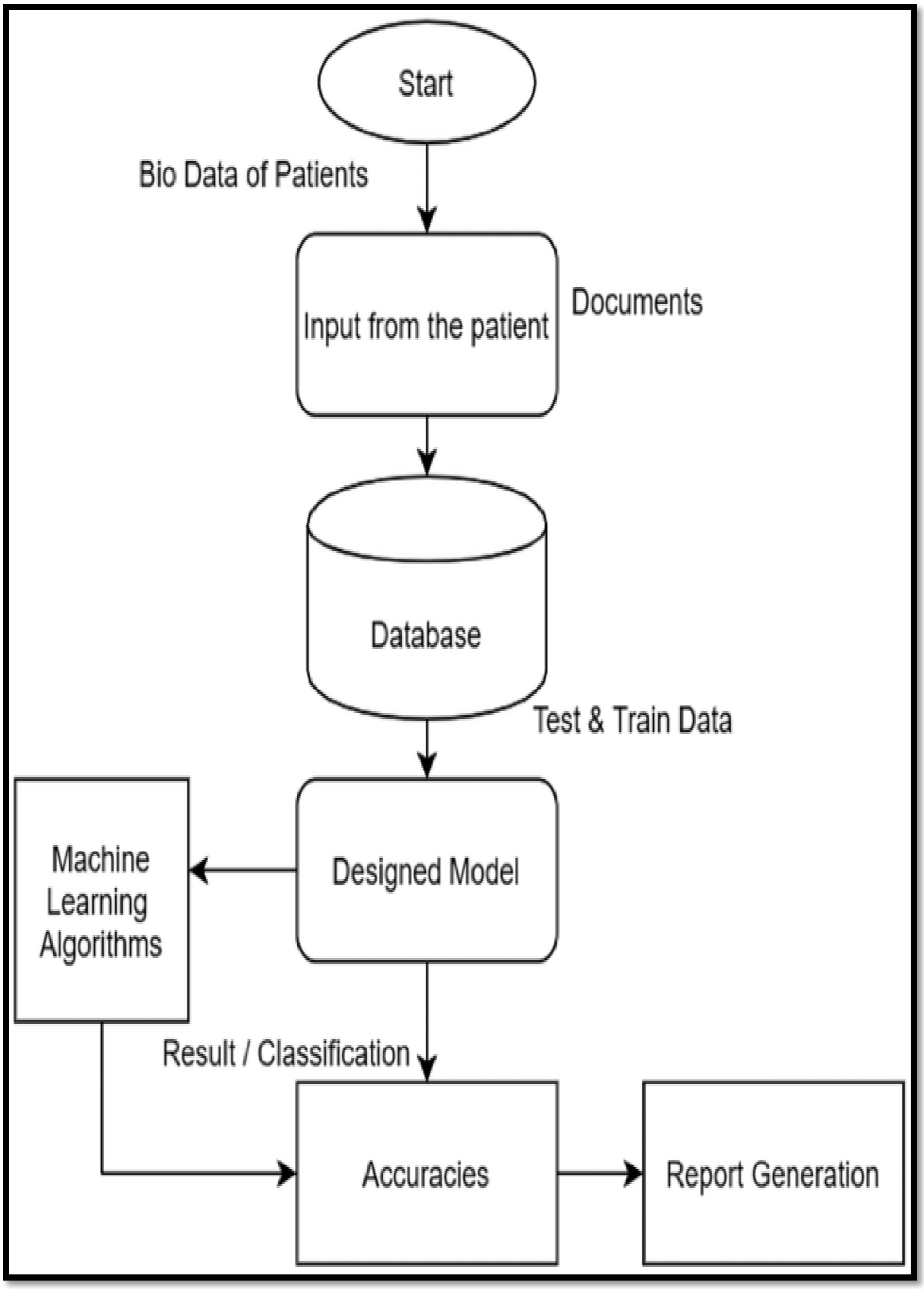
An ANN is based on a collection of connected units or nodes called artificial neurons (analogous to biological neurons in an animal brain). Each connection between artificial neurons can transmit a signal from one to another.

<u>MACHINE LEARNING</u>



MACHINE LEARNING
SUPERVISED LEARNING
Develop predictive model based on both input and output data
UNSUPERVISED LEARNING
Group and interpret data based only on input data
CLASSIFICATION
REGRESSION
CLUSTERING

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed.
Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data.



Flow Chart For Program Methodology

METHODOLOGY

INTRODUCTION TO MACHINE LEARNING

Machine learning is a field of computer science that gives computers the ability to learn without being explicitly programmed.

Arthur Samuel, an American pioneer in the field of computer gaming and artificial intelligence, coined the term "Machine Learning" in 1959 while at IBM. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data

Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning "signal" or "feedback" available to a learning system:-

SUPERVISED LEARNING

Supervised learning is the machine learning task of inferring a function from *labelled training data*.[1] The training data consist of a set of *training examples*. In supervised learning, each example is a *pair* consisting of an input object (typically a vector) and a desired output value.

A supervised learning algorithm analyses the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.

UNSUPERVISED LEARNING

Unsupervised learning is the machine learning task of inferring a function to describe hidden structure from "unlabelled" data (a classification or categorization is not included in the observations). Since the examples given to the learner are unlabelled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm—which is one way of distinguishing unsupervised learning from supervised learning and reinforcement learning.A central case of unsupervised learning is the problem of density estimation in statistics, though unsupervised learning encompasses many other problems (and solutions) involving summarizing and explaining key features of the data

NUMPY

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin.

NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars.

NUMPY ARRAY

NumPy’s main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called *axes*. The number of axes is *rank.*

For example, the coordinates of a point in 3D space [1, 2, 1] is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[1., 0., 0.],
 [ 0., 1., 2.]]
```

NumPy’s array class is called *ndarray*. It is also known by the alias.

SLICING NUMPY ARRAY

Import numpy as np

a = np.array ([[1, 2, 3],[3,4,5],[4,5,6]])

```
print 'Our array is:'
Print a
print '\n'
print 'The items in the second column are:'
print a[...,1]
print '\n'

print 'The items in the second row are:'
print a[1...]
print '\n'

print 'The items columns 1 onwards are:'
print a [...,1:]
```

OUTPUT

```
Our array is:
[[1 2 3]
[3 4 5]
[4 5 6]]

The items in the second column are:
[2 4 5]

The items in the second row are:
[3 4 5]

The items column 1 onwards are:
[[2 3]
[4 5]
[5 6]]
```

SCIPY

Modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.

SciPy builds on the NumPy array object and is part of the NumPy stack which includes tools like Matplotlib, pandas and SymPy, and an expanding set of scientific computing libraries. This NumPy stack has similar users to other applications such as MATLAB, GNU Octave, and Scilab. The NumPy stack is also sometimes referred to as the SciPy stack.

The SciPy Library/Package

The SciPy package of key algorithms and functions core to Python's scientific computing capabilities. Available sub-packages include:

- **constants:** physical constants and conversion factors (since version 0.7.0)
- **cluster:** hierarchical clustering, vector quantization, K-means
- **fftpack:** Discrete Fourier Transform algorithms
- **integrate:** numerical integration routines
- **interpolate:** interpolation tools
- **io:** data input and output
- **lib:** Python wrappers to external libraries
- **linalg:** linear algebra routines
- **misc:** miscellaneous utilities (e.g. image reading/writing)
- **ndimage:** various functions for multi-dimensional image processing
- **optimize:** optimization algorithms including linear programming
- **signal:** signal processing tools
- **sparse:** sparse matrix and related algorithms
- **spatial:** KD-trees, nearest neighbours, distance functions
- **special:** special functions
- **stats:** statistical functions
- **weave:** tool for writing C/C++ code as Python multiline strings

Data Structures

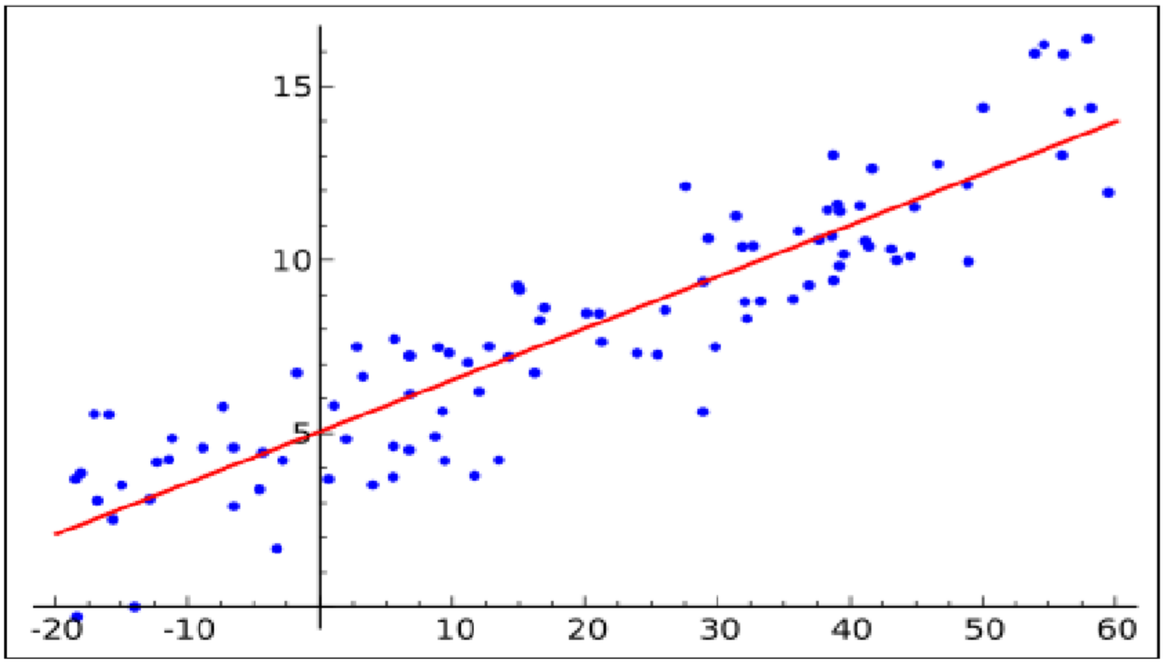
The basic data structure used by SciPy is a multidimensional array provided by the NumPy module. NumPy provides some functions for linear algebra, Fourier transforms and random number generation, but not with the generality of the equivalent functions in SciPy. NumPy can also be used as an efficient multi-dimensional container of data with arbitrary data-types. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases. Older versions of SciPy used Numeric as an array type, which is now deprecated in favour of the newer NumPy array code.

SCIKIT-LEARN

Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits.learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.[4] The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from INRIA took leadership of the project and made the first public release on February the 1st 2010[5]. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012.

REGRESSION ANALYSIS



In statistical modelling, **regression analysis** is a set of statistical processes for estimating the relationships among variables. It includes many techniques for modelling and analysing several variables, when the focus is on the relationship between a dependent variable and one or more independent variables (or 'predictors'). More specifically, regression analysis helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed.

Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer casual relationships between the independent and dependent variables. However this can lead to illusions or false relationships, so caution is advisable

LINEAR REGRESSION

Linear regression is a linear approach for modelling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted X. The case of one explanatory variable is called *simple linear regression*. For more than one explanatory variable, the process is called *multiple linear regression*.

In linear regression, the relationships are modelled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called *linear models*.

LOGISTIC REGRESSION

Logistic regression, or logit regression, or logit model [1] is a regression model where the dependent variable (DV) is categorical. This article covers the case of a binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed in multinomial logistic regression, or, if the multiple categories are ordered, in ordinal logistic regression. In the terminology of economics, logistic regression is an example of a qualitative response/discrete choice model.

POLYNOMIAL REGRESSION

Polynomial regression is a form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modelled as an n^{th} degree polynomial in x.

Polynomial regression fits a nonlinear relationship between the value of x and the corresponding conditional mean of y , denoted E(y | x), and has been used to describe nonlinear phenomena such as the growth rate of tissues, the distribution of carbon isotopes in lake sediments, and the progression of disease epidemics.

Although *polynomial regression* fits a nonlinear model to the data, as a statistical estimation problem it is linear, in the sense that the regression function E(y | x) is linear in the unknown parameters that are estimated from the data.

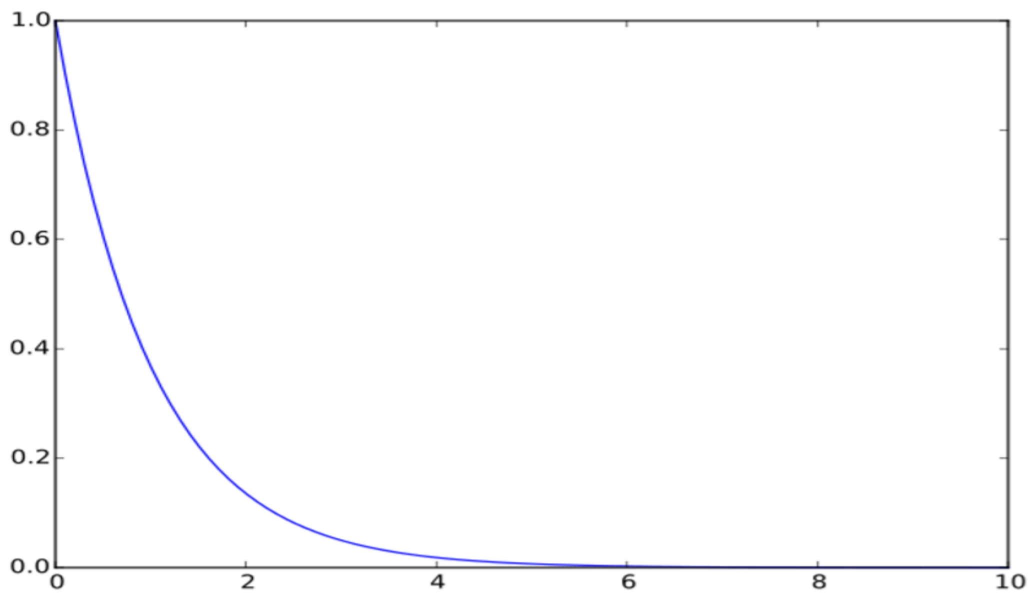
MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter,wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged .SciPy makes use of matplotlib.

EXAMPLE

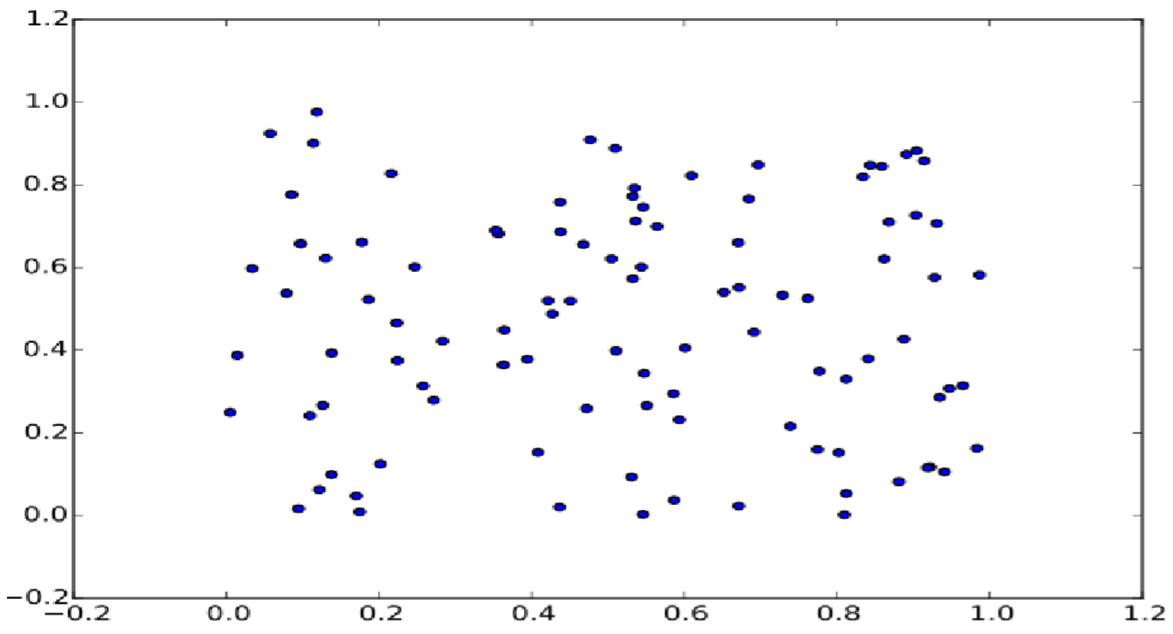
- **LINE PLOT**

```
>>>importmatplotlib.pyplotasplt
>>>importnumpyasnp
>>> a =np.linspace(0,10,100)
>>> b =np.exp(-a)
>>>plt.plot (a,b)
>>>plt.show ()
```



- **SCATTER PLOT**

```
>>>importmatplotlib.pyplotasplt
>>>fromnumpy.randomimport rand
>>> a =rand(100)
>>> b =rand(100)
>>>plt.scatter(a, b)
>>>plt.show ()
```



PANDAS

In computer programming, **pandas** is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. "Panel data", an econometrics term for multidimensional, structured data sets.

LIBRARY FEATURES

- Data Frame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and sub setting of large data sets.

- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation.

CLUSTERING

Cluster analysis or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense or another) to each other than to those in other groups (clusters). It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression, and computer graphics.

Cluster analysis itself is not one specific algorithm, but the general task to be solved. It can be achieved by various algorithms that differ significantly in their notion of what constitutes a cluster and how to efficiently find them. Popular notions of clusters include groups with small distances among the cluster members, dense areas of the data space, intervals or particular statistical

distributions. Clustering can therefore be formulated as a multi-objective optimization problem.

The appropriate clustering algorithm and parameter settings (including values such as the distance function to use, a density threshold or the number of expected clusters) depend on the individual data set and intended use of the results. Cluster analysis as such is not an automatic task, but an iterative process of knowledge discovery or interactive multi-objective optimization that involves trial and failure. It is often necessary to modify data pre-processing and model parameters until the result achieves the desired properties.

ALGORITHM

- Data Collection
- Data Formatting
- Model Selection
- Training
- Testing

Data Collection: We have collected data sets of weather from online website. We have downloaded the .csv files in which information was present.

Data Formatting: The collected data is formatted into suitable data sets. We check the collinearity with mean temperature. The data sets which have collinearity nearer to 1.0 has been selected.

Model Selection: We have selected different models to minimize the error of the predicted value. The different models used are Linear Regression Linear Model, Ridge Linear model, Lasso Linear Model and Bayesian Ridge Linear Model.

Training: The data set was divided such that x_train is used to train the model with corresponding x_test values and some y_train kept reserved for testing.
Testing: The model was tested with y_train and stored in y_predict. Both y_train and y_predict was compared.

Heart Disease Prediction

Introduction: Heart disease puts an enormous burden on the global healthcare system. If the problem is attended to on time, many adverse events could be prevented. Machine Learning holds promise in diagnosing the possibility of adverse cardiac outcomes in high-risk individuals.

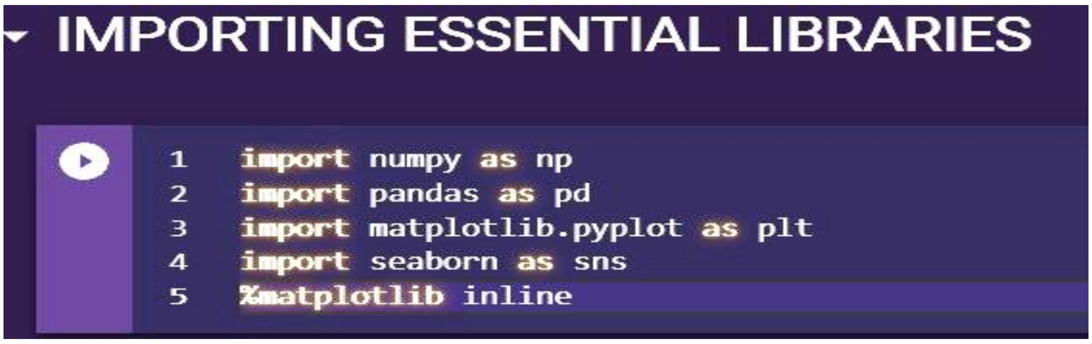
In this article, we will be closely working with the heart disease prediction and for that, we will be looking into the heart disease dataset from that dataset we will derive various insights that help us know the weightage of each feature and how they are interrelated to each other but this time our sole aim is to detect the probability of person that will be affected by a saviour heart problem or not.

Exaplanation:

Takeaways

The Heart Disease prediction will have the following key takeaways:

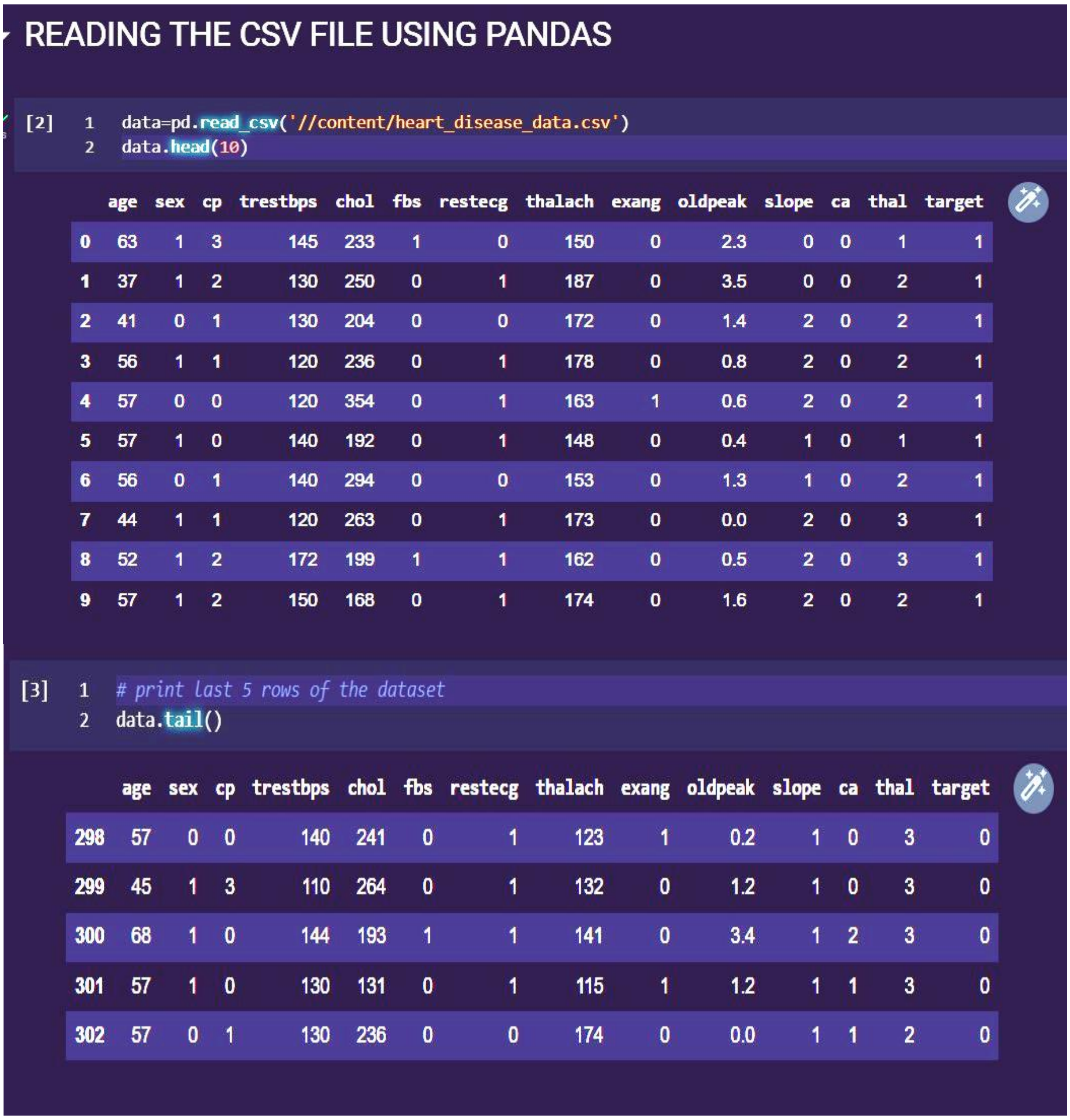
1. **Data insight:** As mentioned here we will be working with the heart disease detection dataset and we will be putting out interesting inferences from the data to derive some meaningful results.
2. **EDA:** Exploratory data analysis is the key step for getting meaningful results.
3. **Model building:** In this phase, we will be building our Machine learning model for heart disease detection.


IMPORTING ESSENTIAL LIBRARIES

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

RESULTS AND DISCUSSIONS

- At first we importing the libraries.
- Then we read the csv file using Pandas library. And print the first 10 rows of dataset and the last 5 rows of dataset.



READING THE CSV FILE USING PANDAS

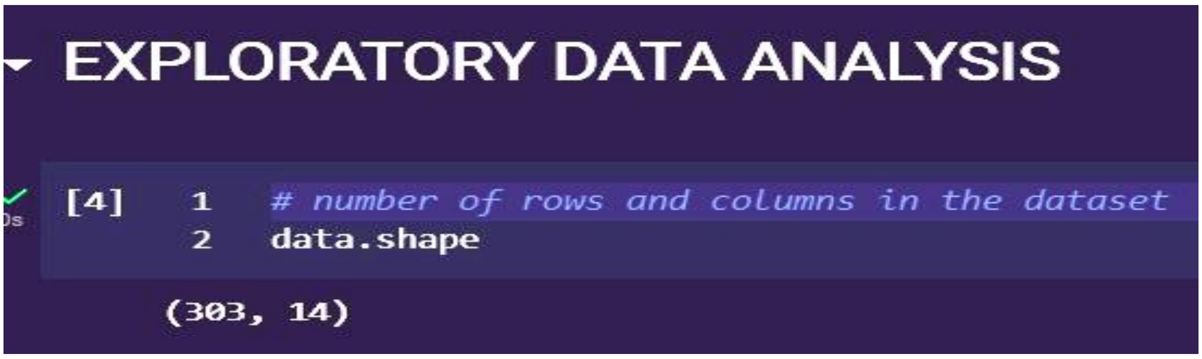
[2]
```python
data=pd.read_csv('//content/heart_disease_data.csv')
data.head(10)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
7	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1
8	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
9	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1

[3]
```python
# print last 5 rows of the dataset
data.tail()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

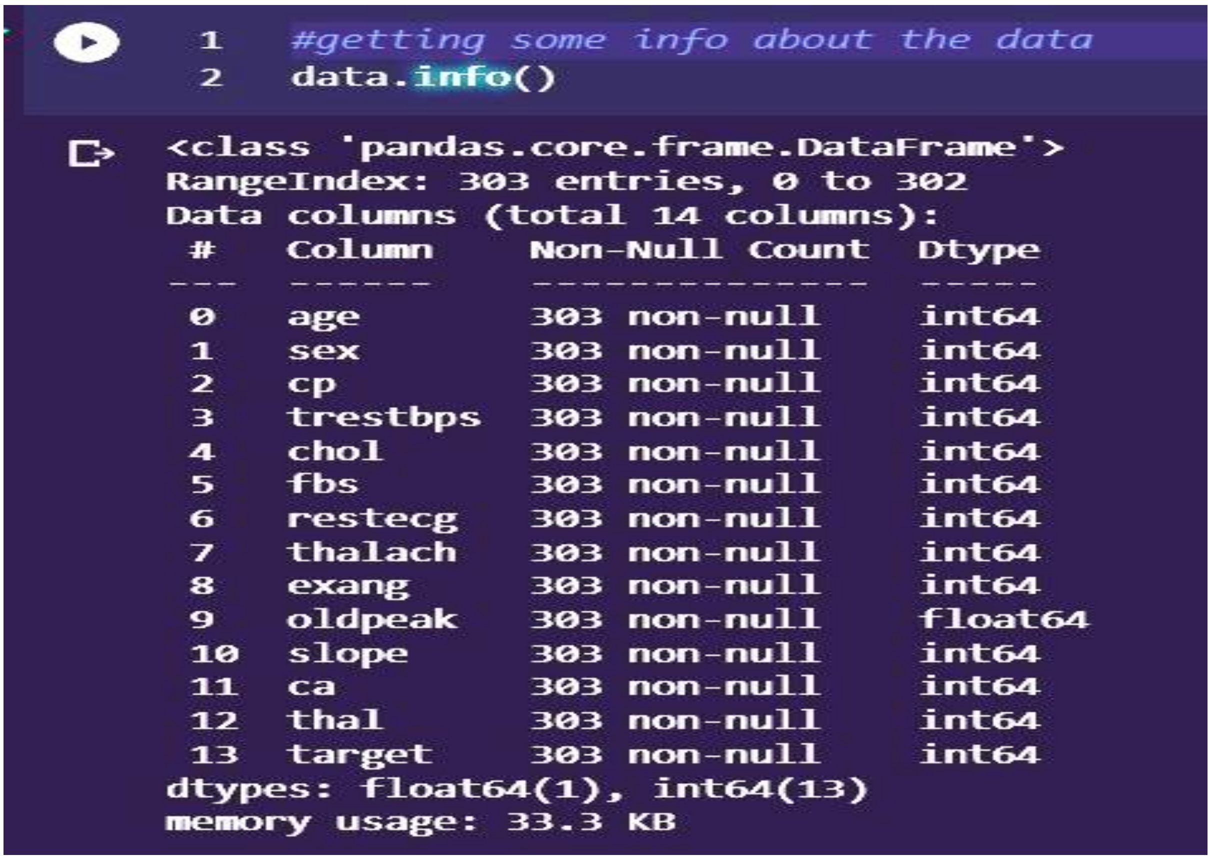
- Let’s analyse the dataset to get a clear insight.



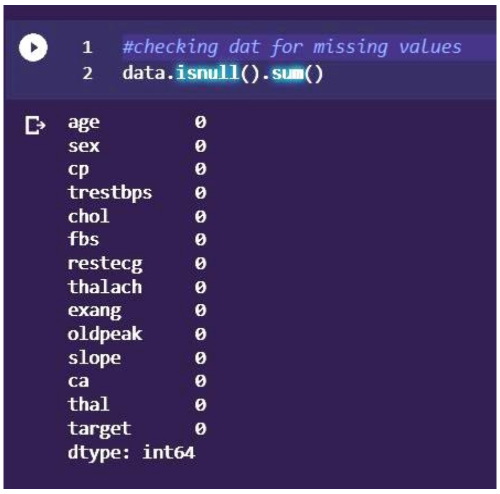
EXPLORATORY DATA ANALYSIS

```
[4]  # number of rows and columns in the dataset
     data.shape

(303, 14)
```

➢ The dataset have 303 rows and 14 columns.



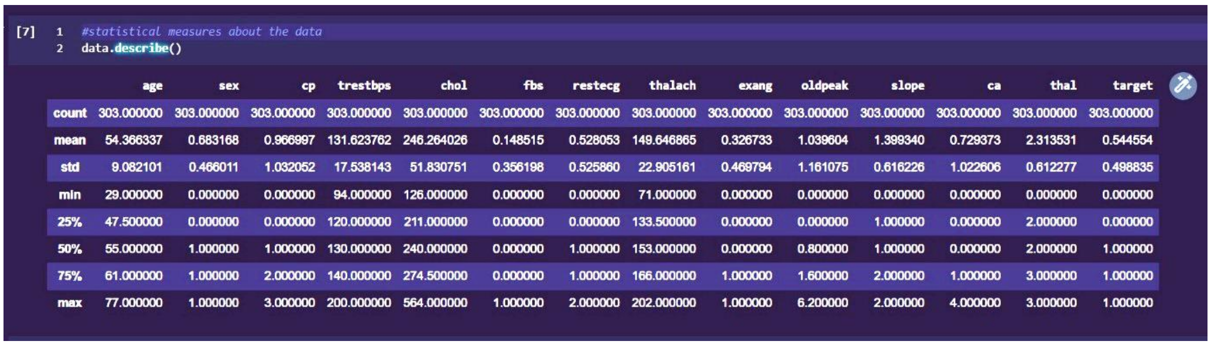
```
#getting some info about the data
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```



```
#checking dat for missing values
data.isnull().sum()

age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

➢ Here we see that no null values are present in the dataset. And the datatype of values also.

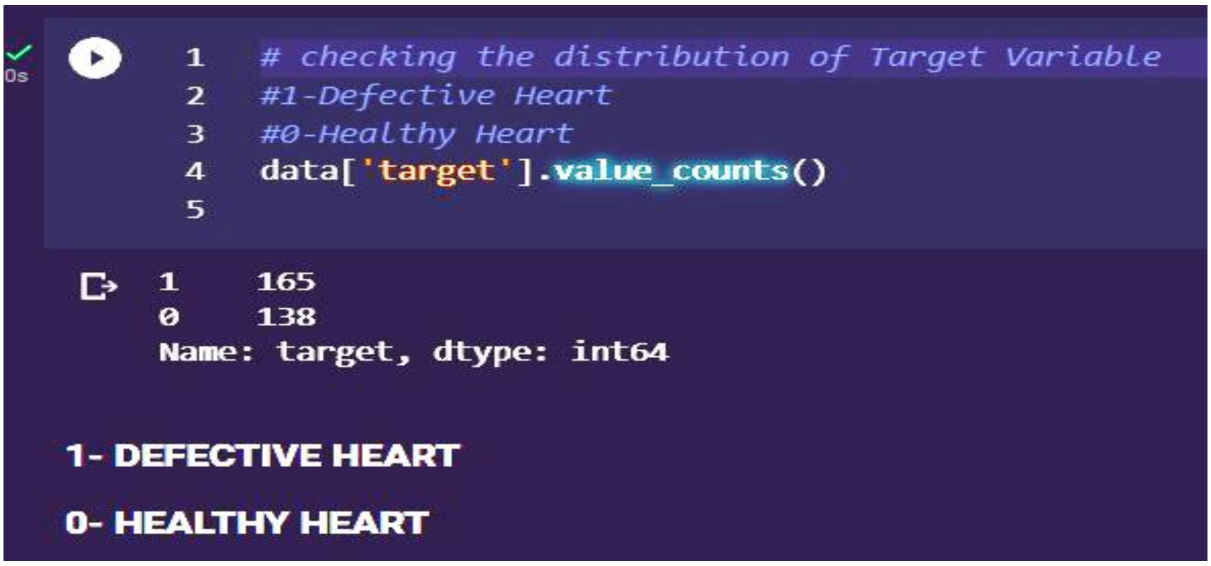
➢ Now let’s see the statistical measure of the dataset by using command “data.describe()”.



```
[7]
#statistical measures about the data
data.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.313531	0.544554
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.612277	0.498835
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

- ➢ The describe() method is used for calculating some statistical data like percentile, mean and std of the numerical values of the Series or Dataframe.



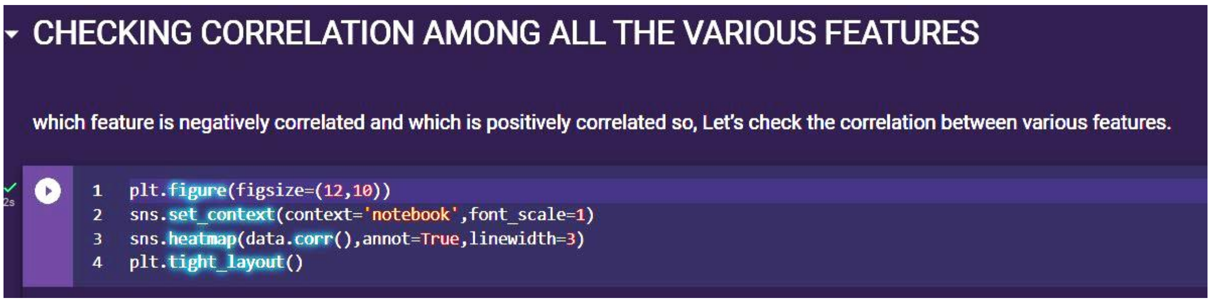
```
# checking the distribution of Target Variable
#1-Defective Heart
#0-Healthy Heart
data['target'].value_counts()

```

```
1    165
0    138
Name: target, dtype: int64
```

1- DEFECTIVE HEART

0- HEALTHY HEART

- ➢ The value_counts() method returns the object containing counts of unique values in sorted order.
- ➢ Here we see the unique values of target in sorted order.
  Here we clearly see that 165 have defective heart and 138 have a healthy heart.

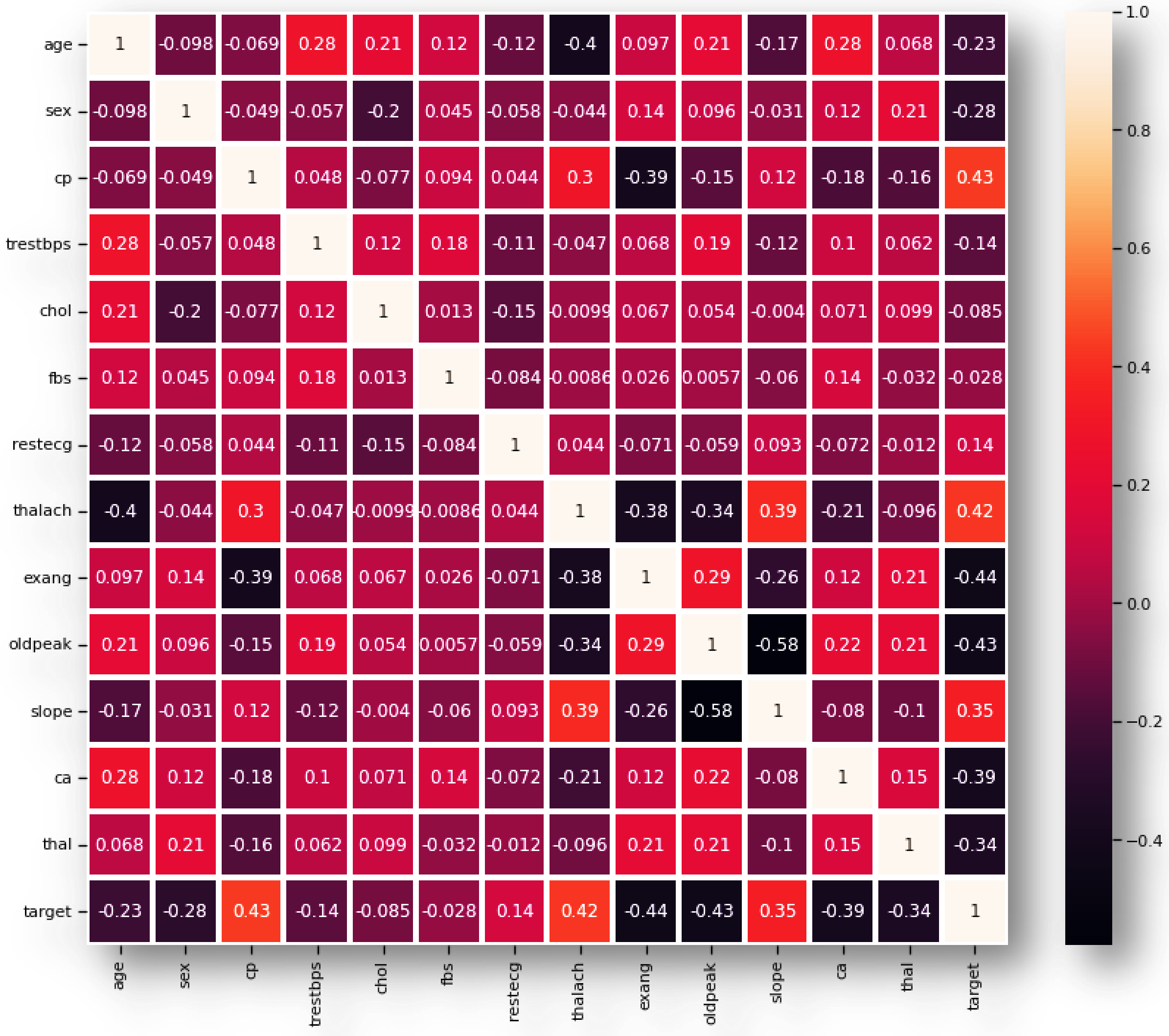


CHECKING CORRELATION AMONG ALL THE VARIOUS FEATURES

which feature is negatively correlated and which is positively correlated so, Let's check the correlation between various features.

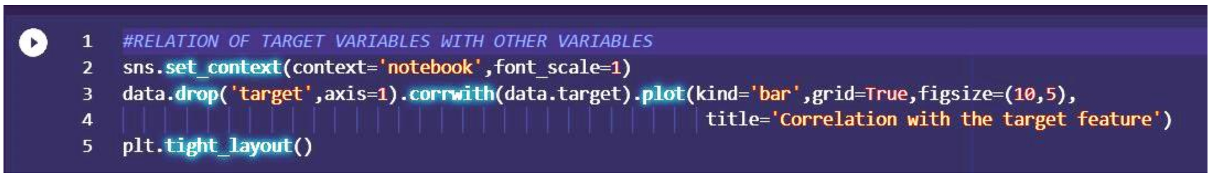
```
plt.figure(figsize=(12,10))
sns.set_context(context='notebook',font_scale=1)
sns.heatmap(data.corr(),annot=True,linewidth=3)
plt.tight_layout()
```

- ➢ Now we checking the correlation among the all features.
  And plot the graphical picture of that using seaborn.

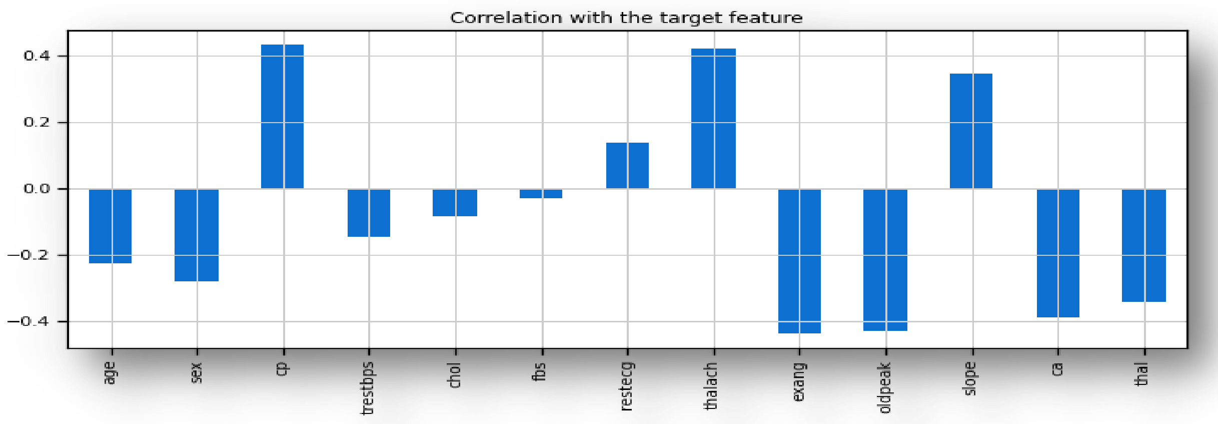


➢ Above graphical representation shows which feature is negatively correlated and which is positively correlated.

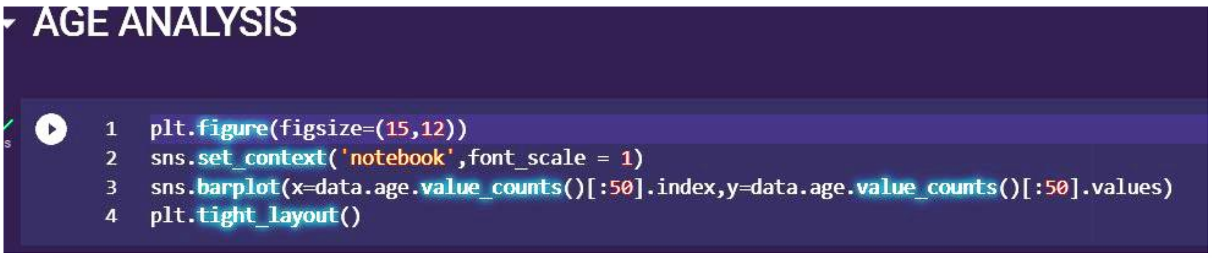
- By far we have checked the correlation between the features but it is also a good practice to check the correlation of the target variable.
- Insights from the above graph are: Four feature(“cp”, “restecg”, “thalach”, “slope”) are positively correlated with the target feature. Other features are negatively correlated with the target feature.
- So, we have done enough collective analysis now let’s go for the analysis of the individual features.



```
#RELATION OF TARGET VARIABLES WITH OTHER VARIABLES
sns.set_context(context='notebook',font_scale=1)
data.drop('target',axis=1).corrwith(data.target).plot(kind='bar',grid=True,figsize=(10,5),
                                                       title='Correlation with the target feature')
plt.tight_layout()
```

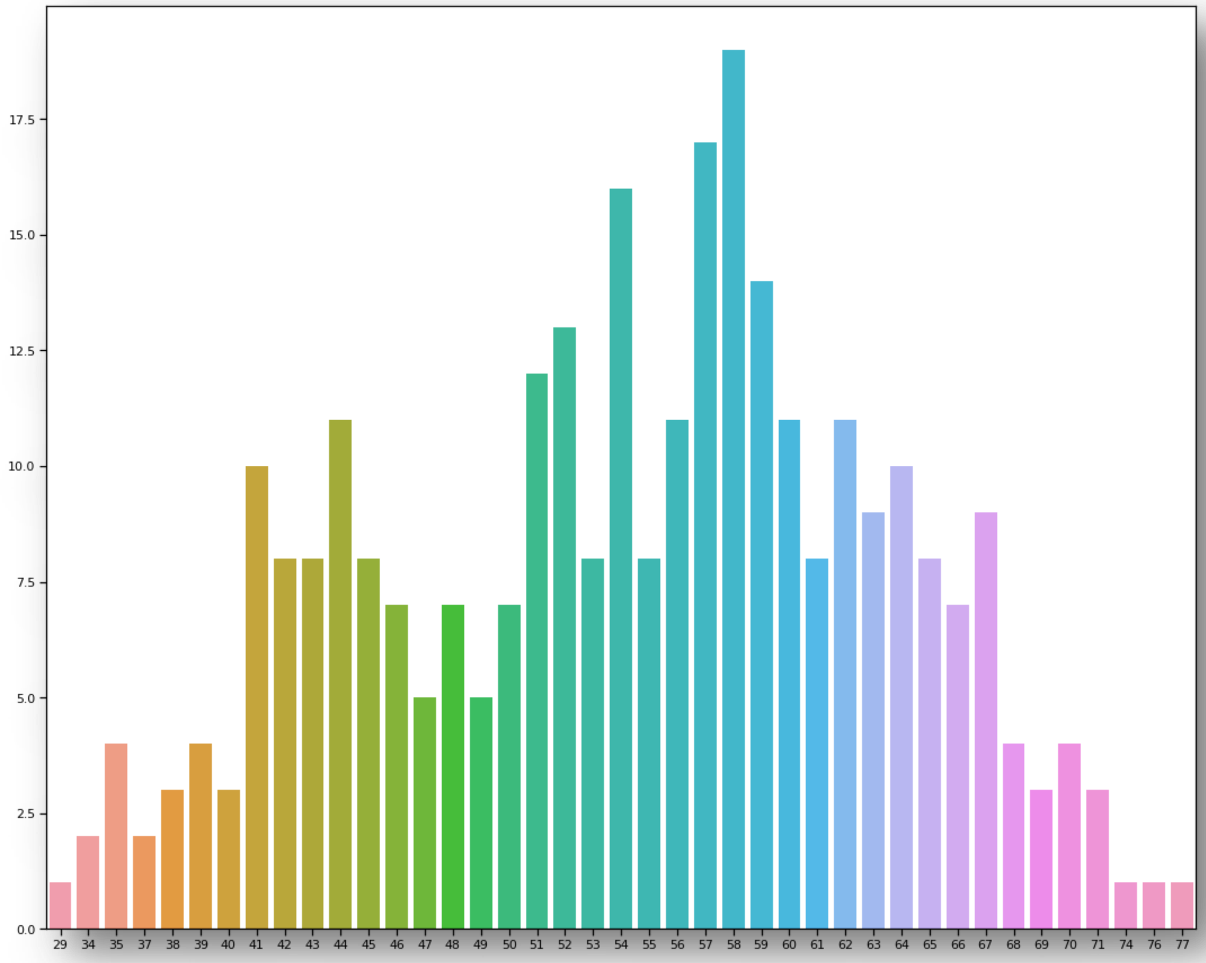


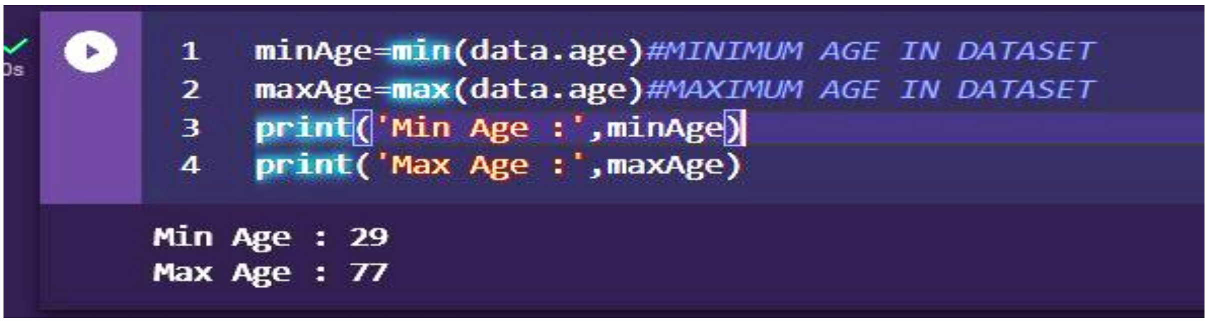
- Now do the age analysis. Here we will be checking the 50 ages and their counts.
- Here we can see that the 58 age column has the highest frequency.
- Let’s check the range of age in the dataset.



AGE ANALYSIS

```
plt.figure(figsize=(15,12))
sns.set_context('notebook',font_scale = 1)
sns.barplot(x=data.age.value_counts()[:50].index,y=data.age.value_counts()[:50].values)
plt.tight_layout()
```

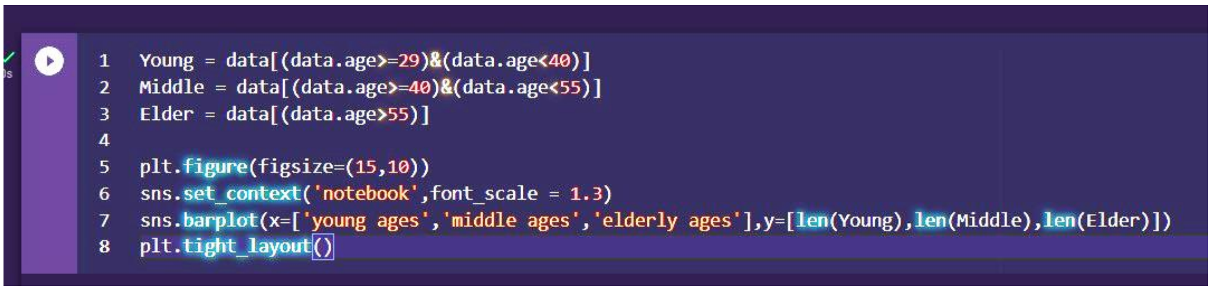




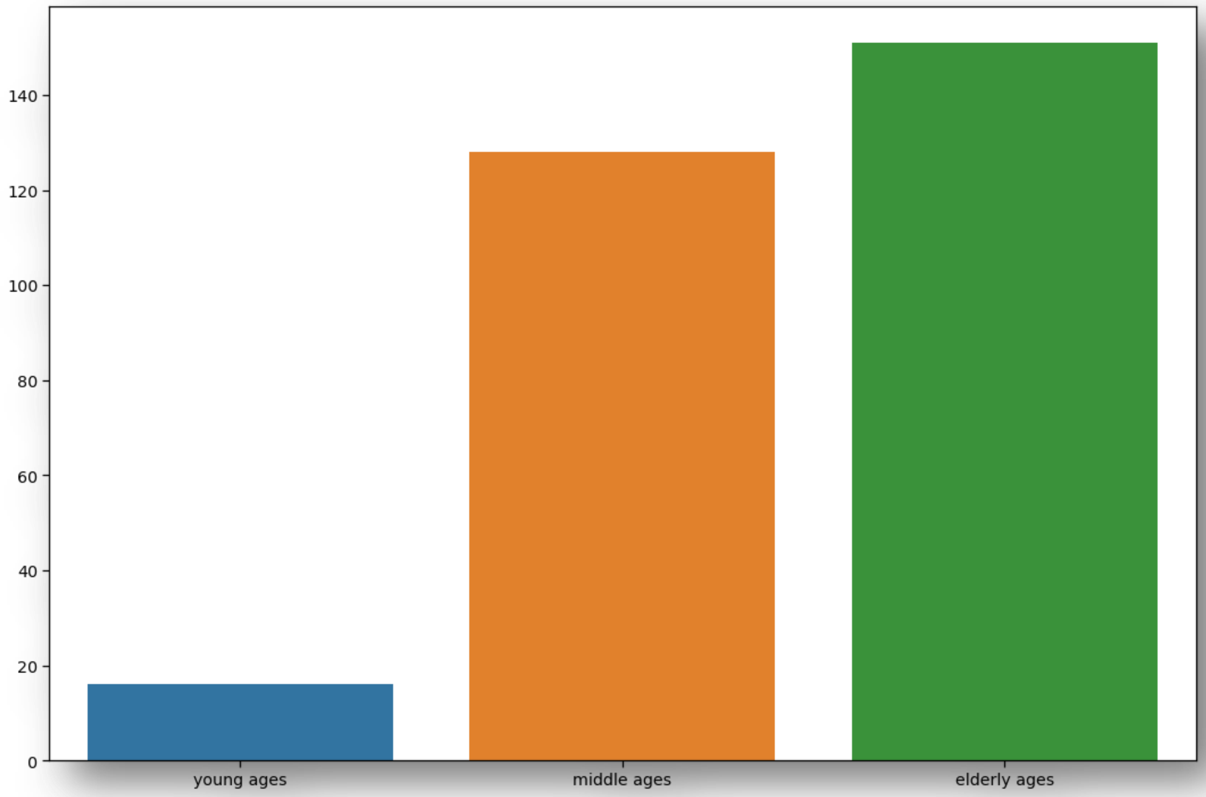
```
minAge=min(data.age)#MINIMUM AGE IN DATASET
maxAge=max(data.age)#MAXIMUM AGE IN DATASET
print('Min Age :',minAge)
print('Max Age :',maxAge)
```

```
Min Age : 29
Max Age : 77
```

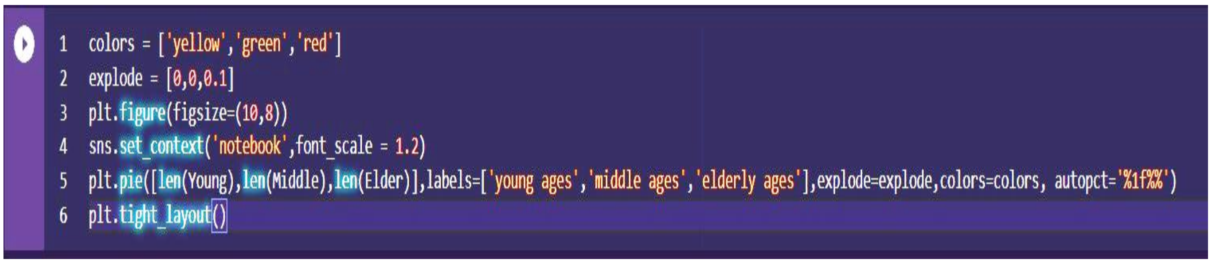
- The minimum age is 29 and maximum age is 77.
- Divide the Age feature into three parts – “Young”, “Middle” and “Elder”.



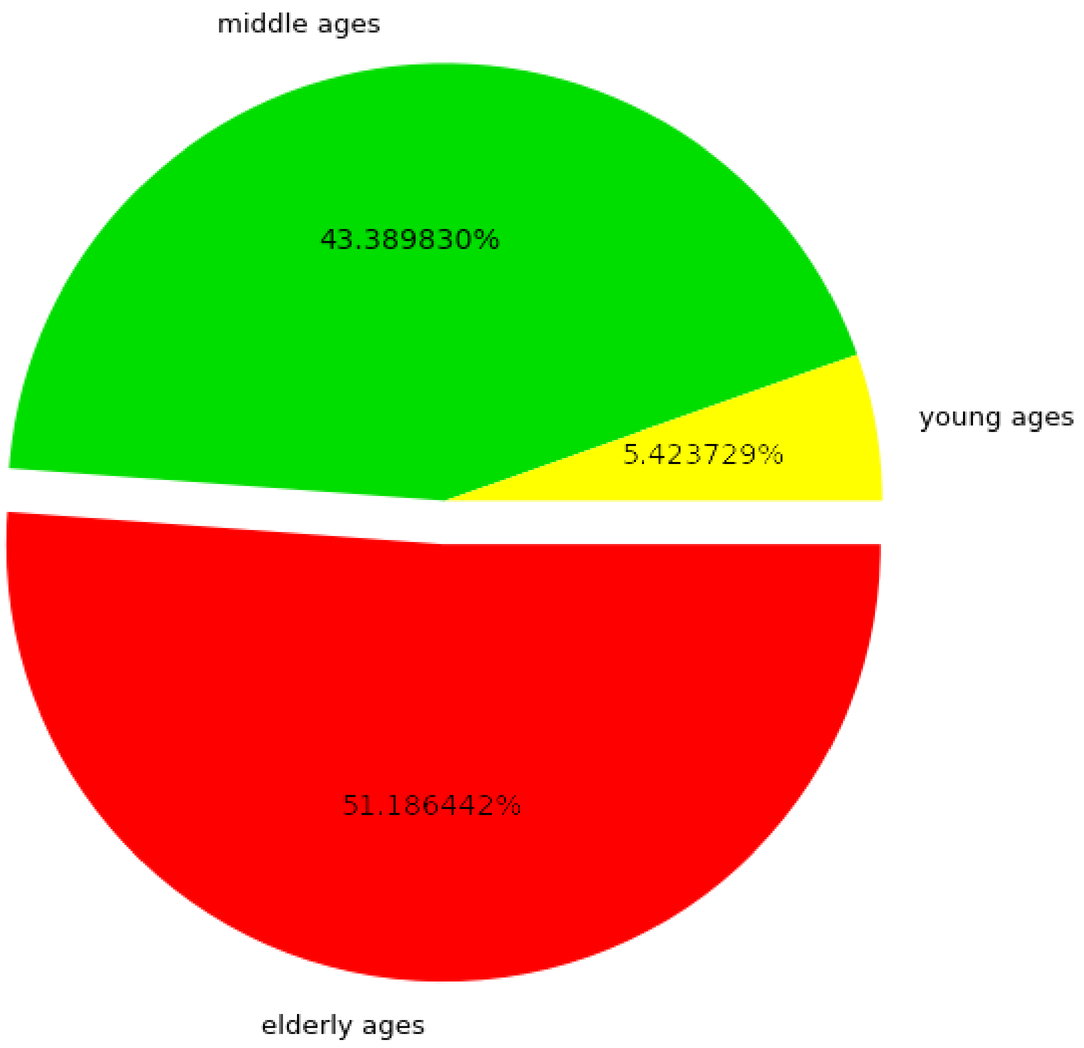
```
Young = data[(data.age>=29)&(data.age<40)]
Middle = data[(data.age>=40)&(data.age<55)]
Elder = data[(data.age>55)]

plt.figure(figsize=(15,10))
sns.set_context('notebook',font_scale = 1.3)
sns.barplot(x=['young ages','middle ages','elderly ages'],y=[len(Young),len(Middle),len(Elder)])
plt.tight_layout()
```



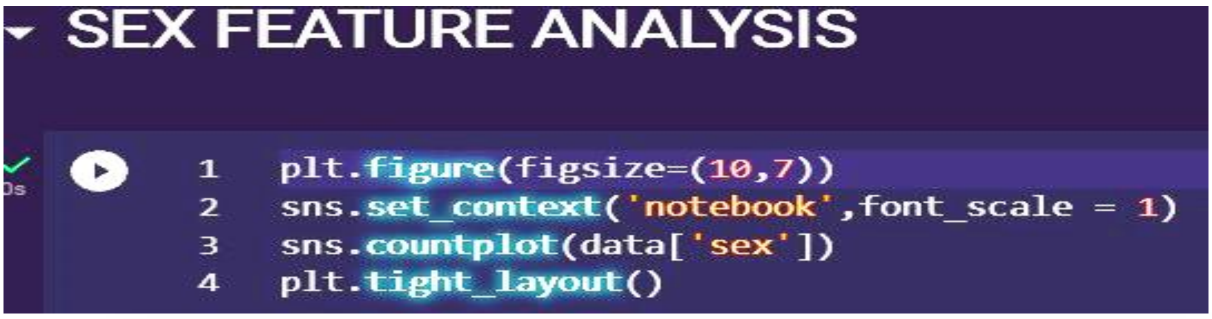
- Above picture shows that elder people are the most affected by heart disease and young ones are the least affected.
- To prove the above inference we will plot the pie chart.



```python
colors = ['yellow','green','red']
explode = [0,0,0.1]
plt.figure(figsize=(10,8))
sns.set_context('notebook',font_scale = 1.2)
plt.pie([len(Young),len(Middle),len(Elder)],labels=['young ages','middle ages','elderly ages'],explode=explode,colors=colors, autopct='%1f%%')
plt.tight_layout()
```

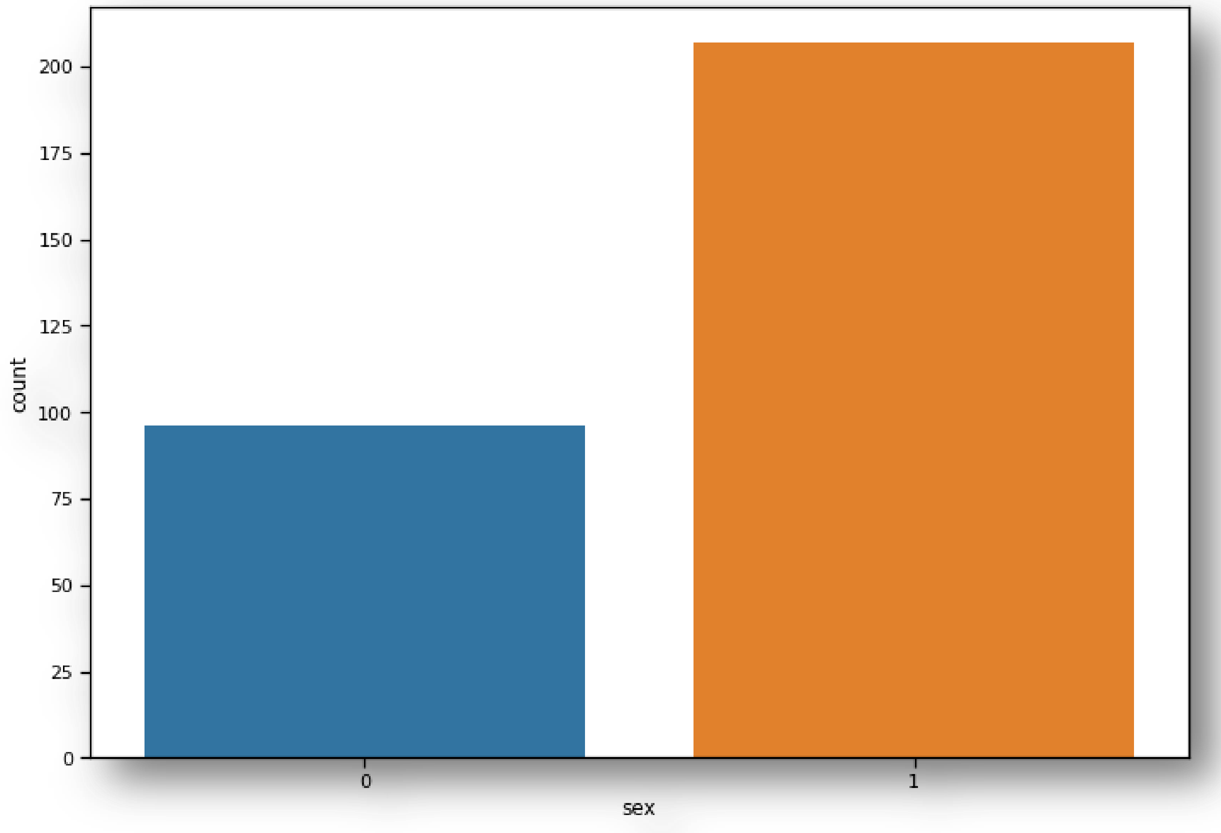


- The pie chart gives the more clear view about the ages.
- Now let’s analyse sex features.

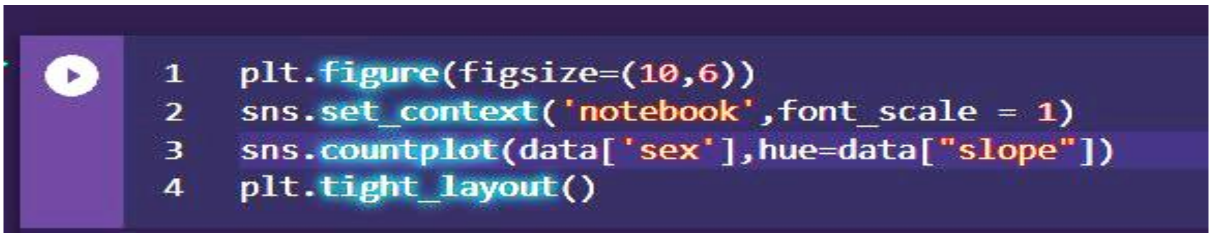


SEX FEATURE ANALYSIS

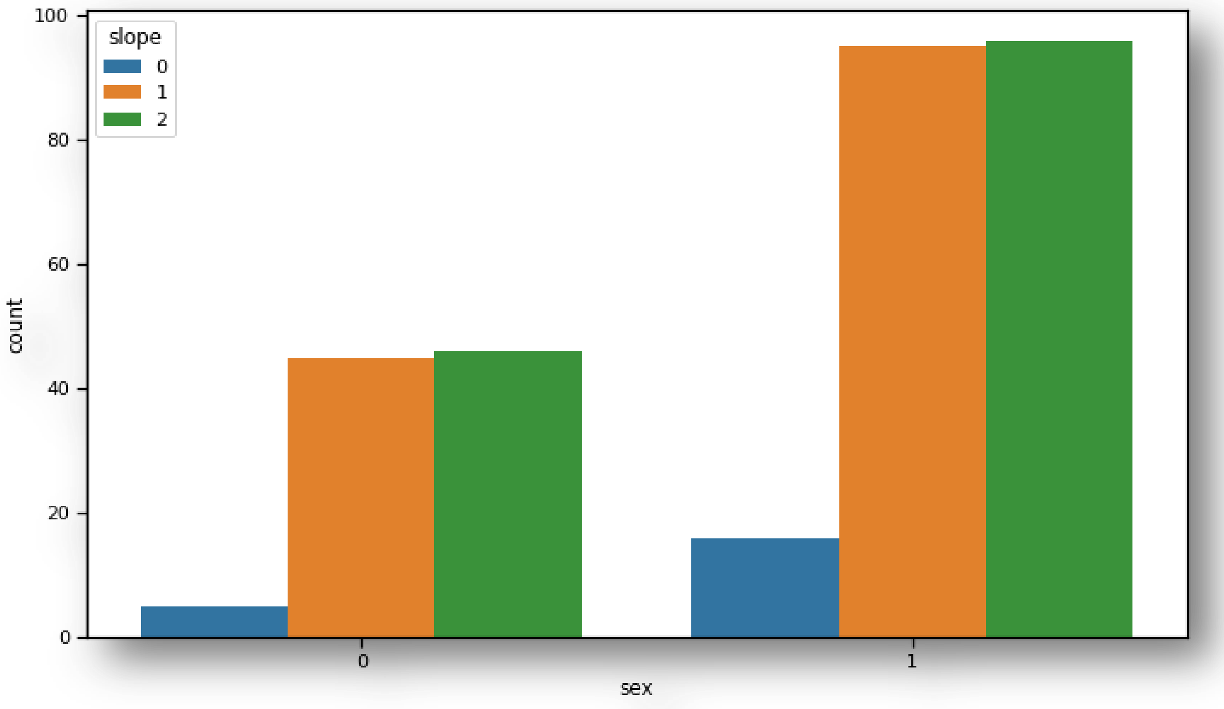
```
plt.figure(figsize=(10,7))
sns.set_context('notebook',font_scale = 1)
sns.countplot(data['sex'])
plt.tight_layout()
```



- Here it is clearly visible that, Ratio of Male to Female is approx 2:1. Here **1- Male** and **0- Female.**
- Now let’s plot the relation between sex and slope.
- slope: the slope of the peak exercise ST segment — 0: downsloping; 1: flat; 2: upsloping 0: downsloping; 1: flat; 2: upsloping

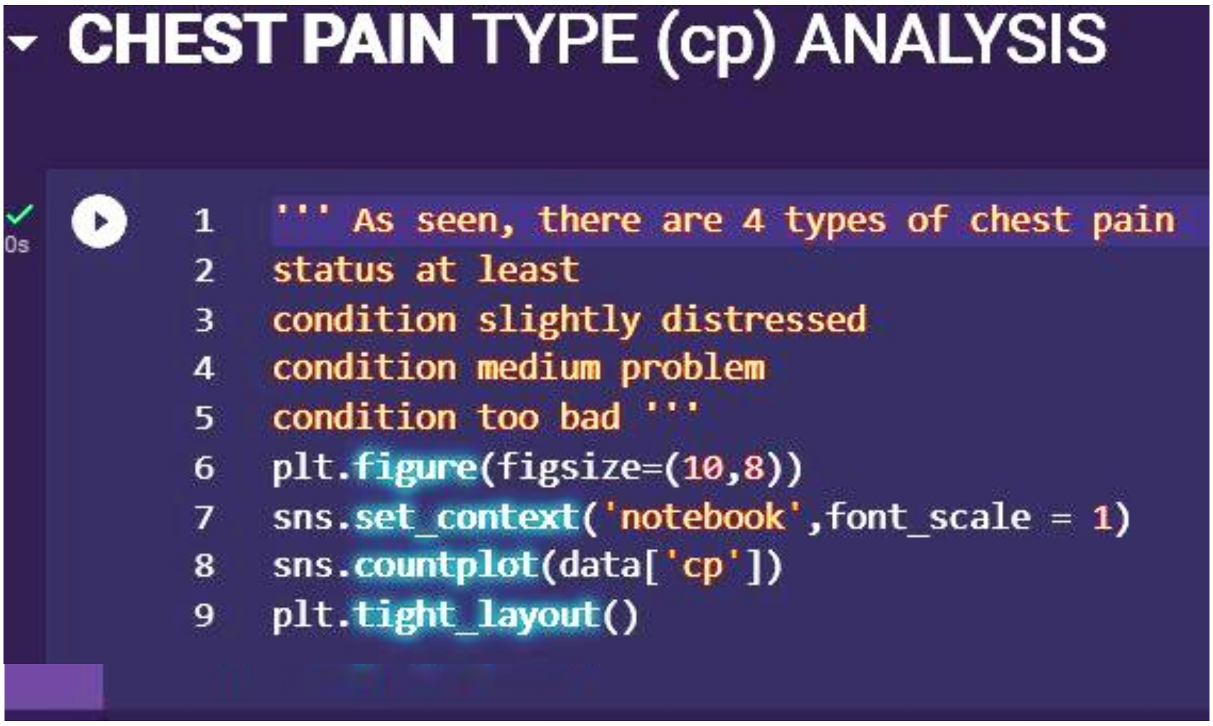


```
plt.figure(figsize=(10,6))
sns.set_context('notebook',font_scale = 1)
sns.countplot(data['sex'],hue=data["slope"])
plt.tight_layout()
```



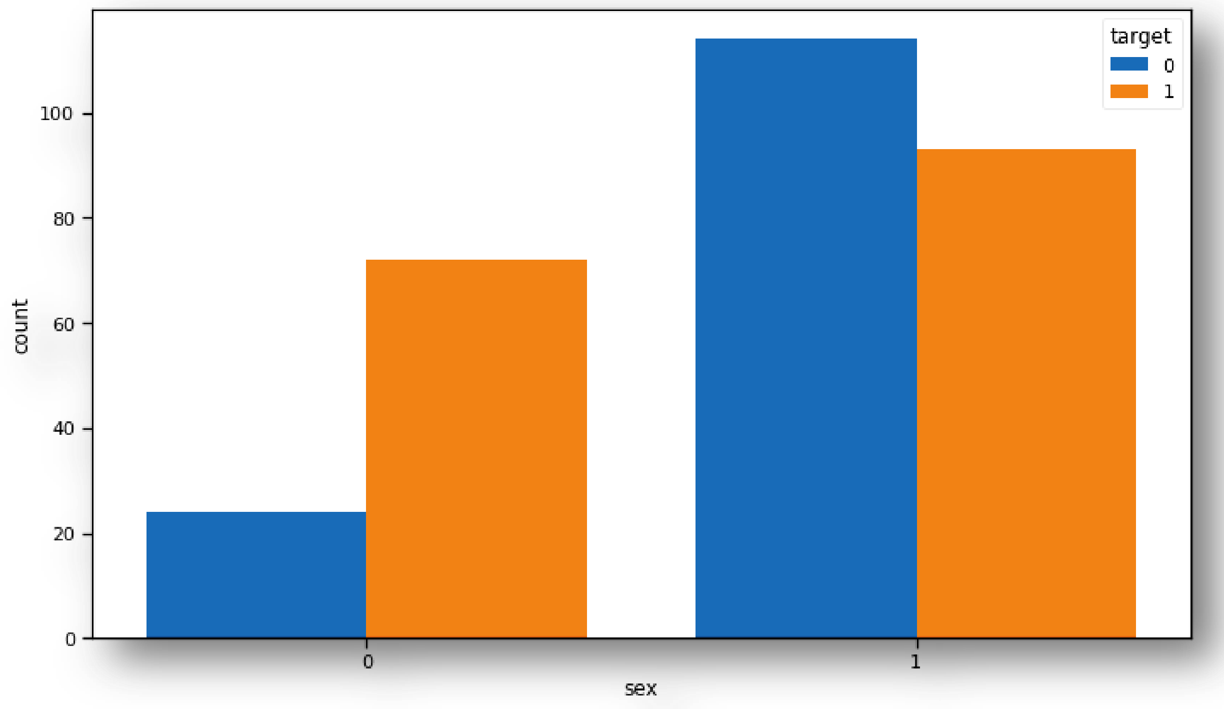
- Here it is clearly visible that the slope value is higher in the case of Males(1).

- Now let’s analyse the sex vs target.
- According to this dataset males are more susceptible to get Heart Disease than females. Men experience heart attacks more than women. Sudden Heart Attacks are experienced by men between 70% — 89%. But if we see the only the male data it shows that maximum male has not heart disease but in case of female it is opposite.
- Let’s analyse the chest pain.

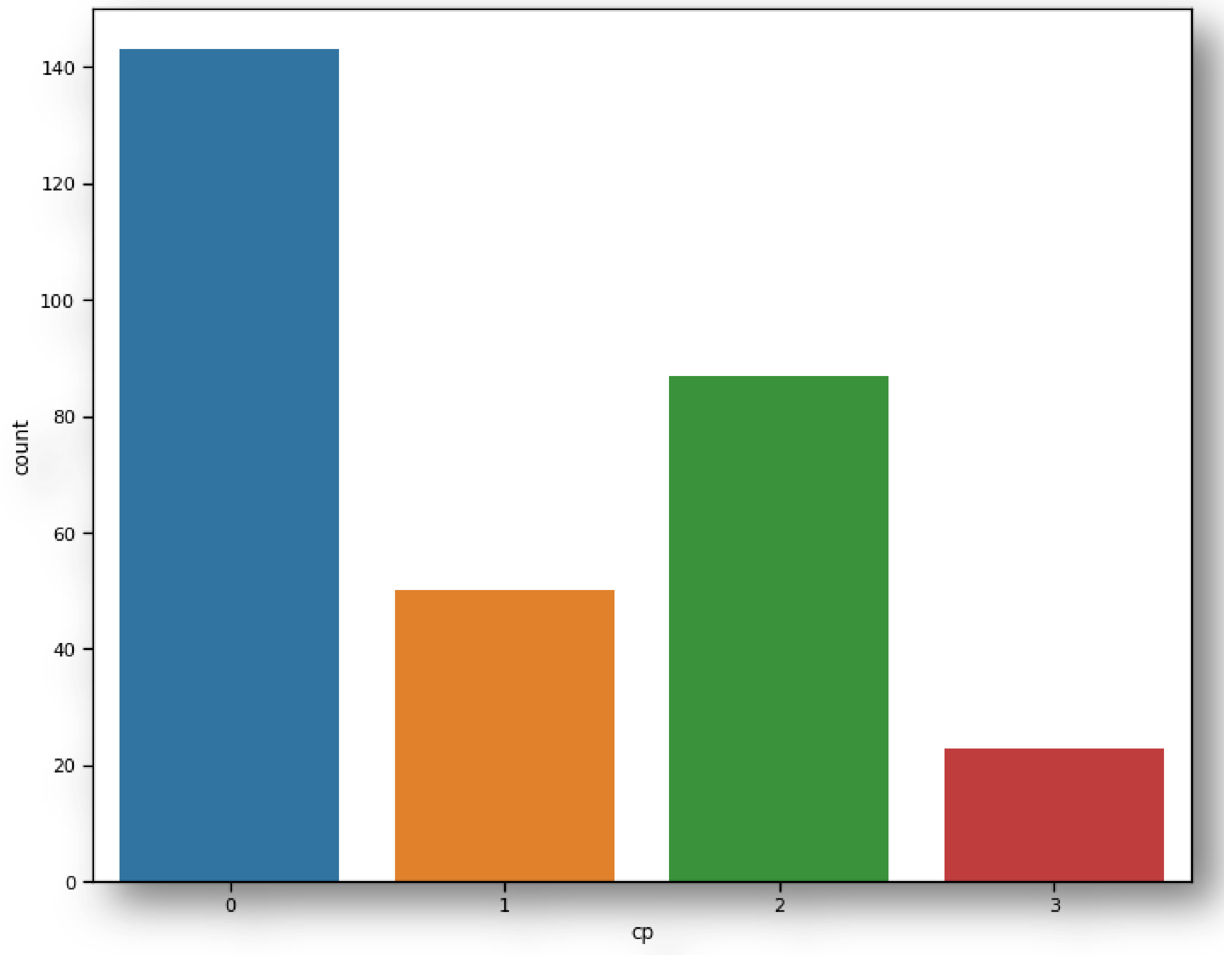


CHEST PAIN TYPE (cp) ANALYSIS

```
''' As seen, there are 4 types of chest pain
status at least
condition slightly distressed
condition medium problem
condition too bad '''
plt.figure(figsize=(10,8))
sns.set_context('notebook',font_scale = 1)
sns.countplot(data['cp'])
plt.tight_layout()
```

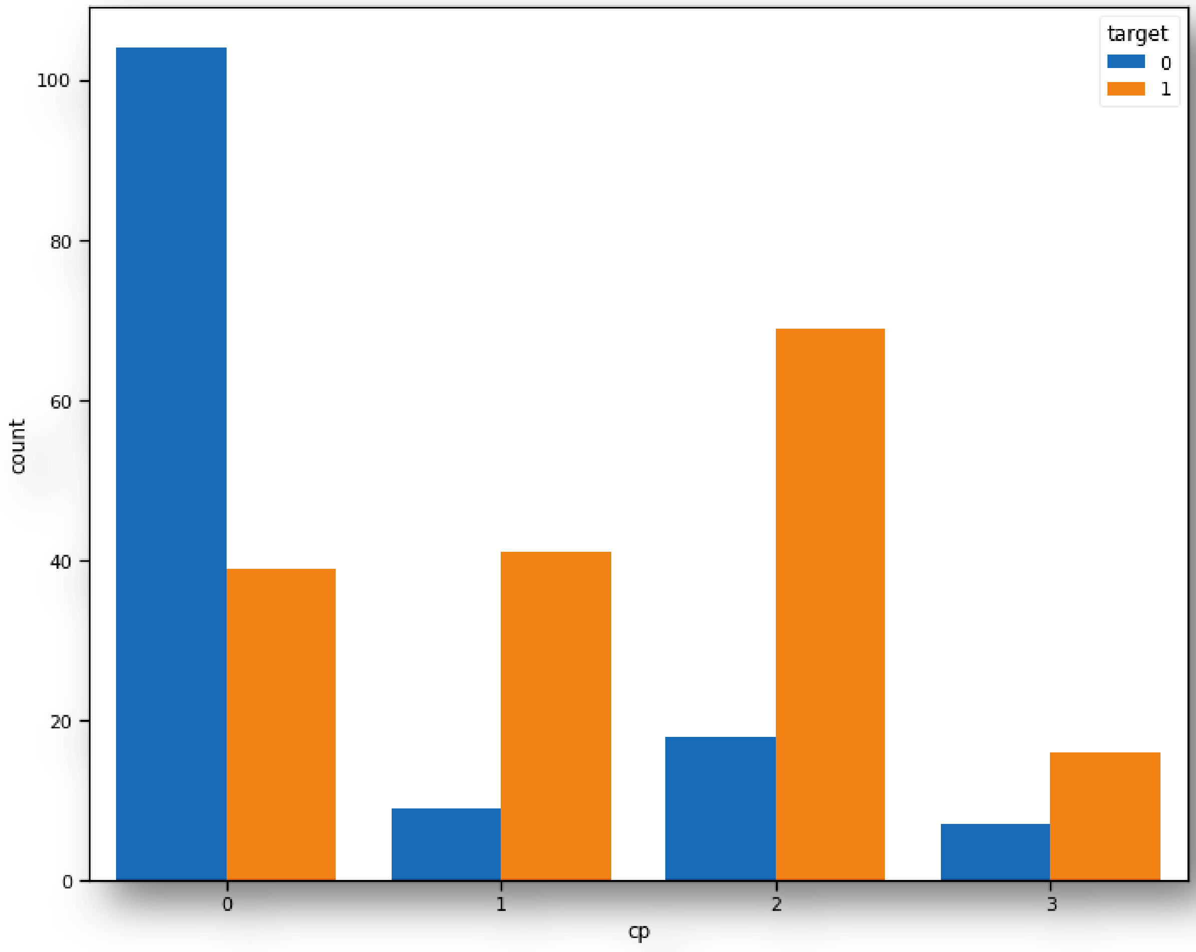


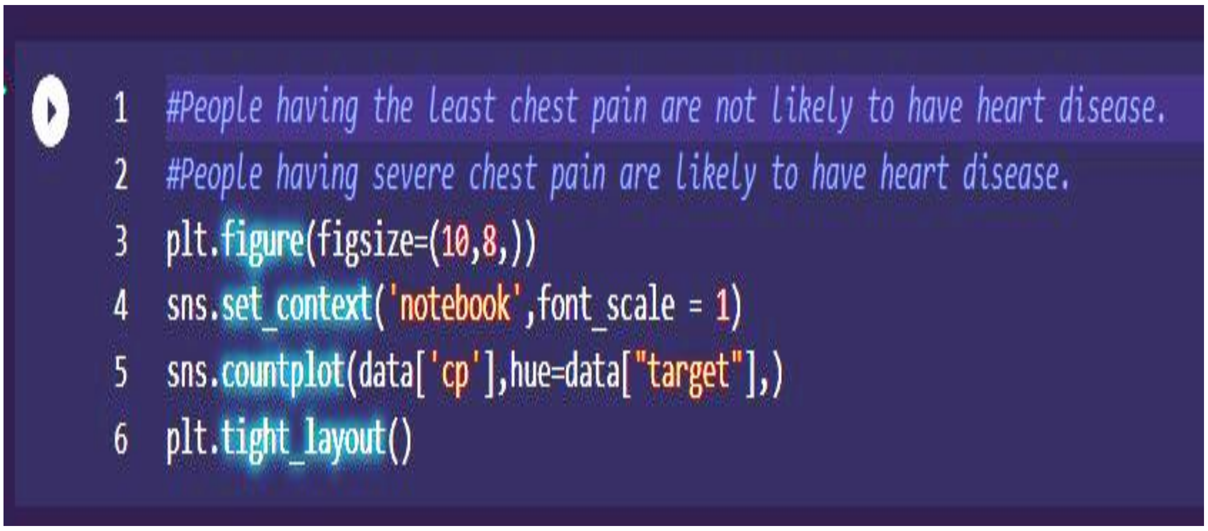
target
0
1
count
sex
0
1
0
20
40
60
80
100



count
cp
0
1
2
3
0
20
40
60
80
100
120
140

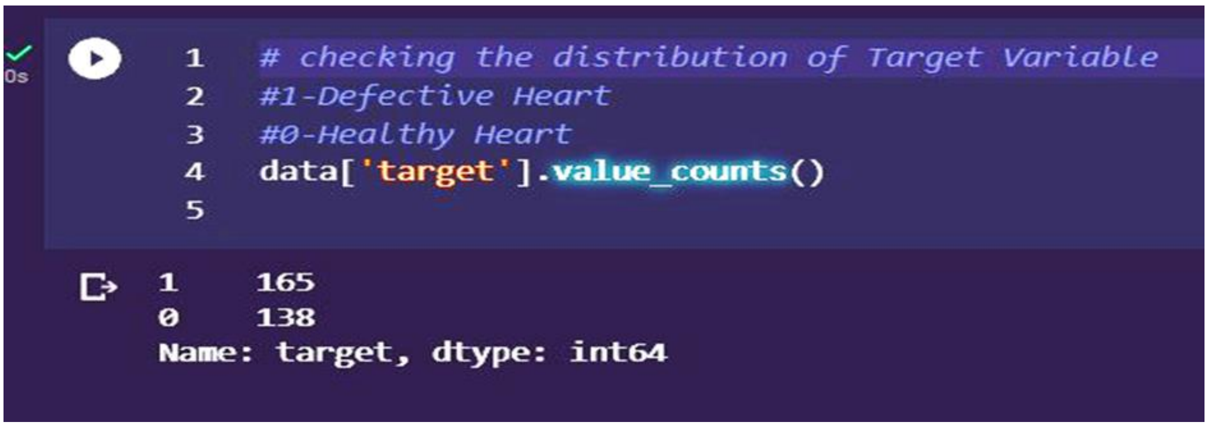
- As seen, there are 4 types of chest pain.
  1. status at least.
  2. condition slightly distressed
  3. condition medium problem
  4. condition too bad.
- Let’s analyse CP vs Target.





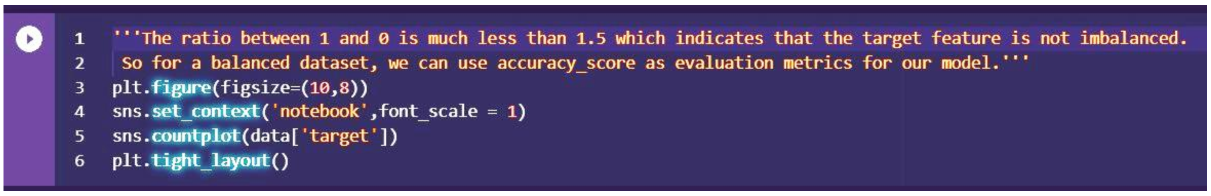
```
#People having the least chest pain are not likely to have heart disease.
#People having severe chest pain are likely to have heart disease.
plt.figure(figsize=(10,8,))
sns.set_context('notebook',font_scale = 1)
sns.countplot(data['cp'],hue=data["target"],)
plt.tight_layout()
```

- The graph shows that when chest pain are increasing the risk of heart failure is high.
- Let’s see the Target of balanced dataset.
- The ratio between 1 and 0 is much less than 1.5 which indicates that the target feature is not imbalanced. So for a balanced dataset, we can use accuracy_score as evaluation metrics for our model.



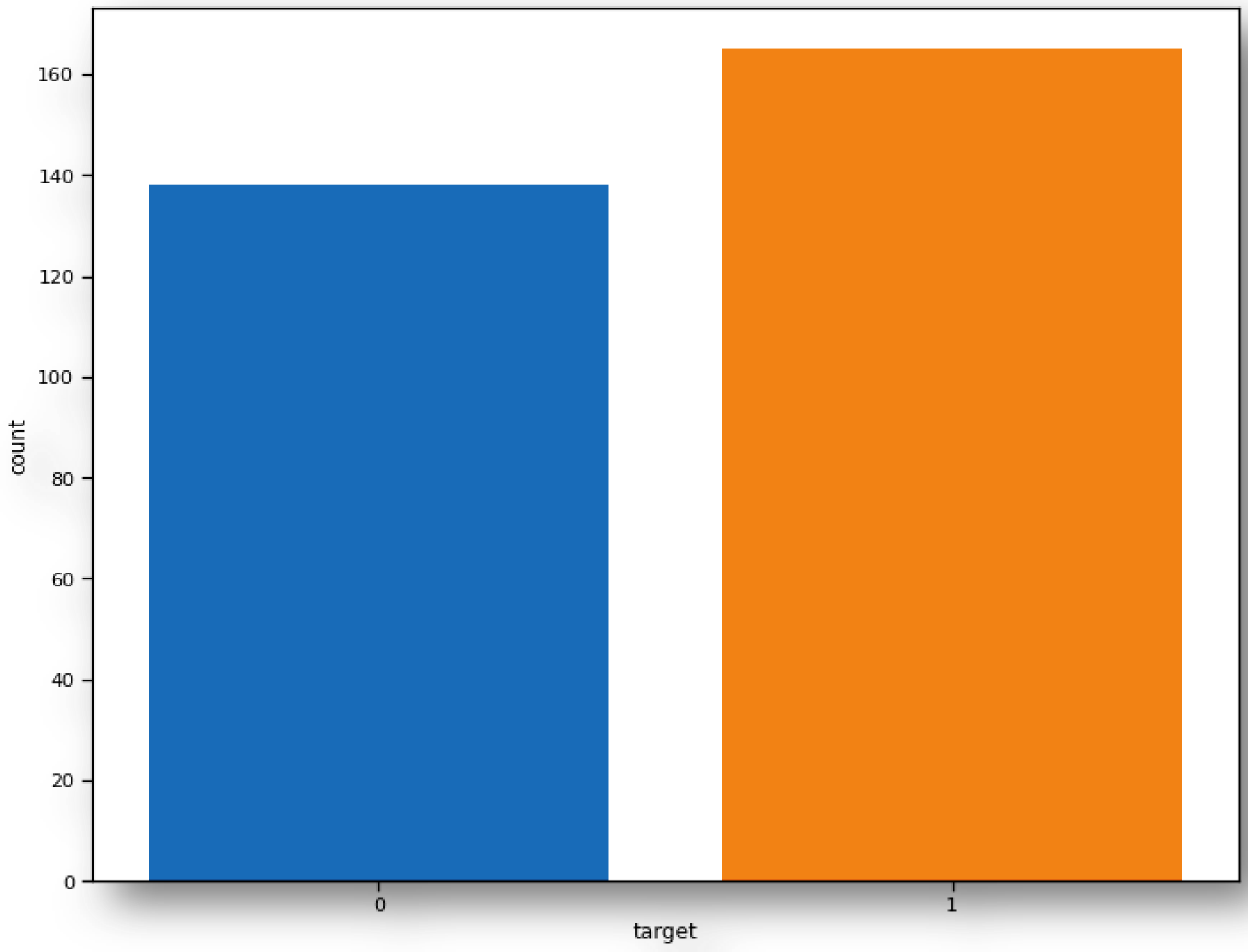
```
# checking the distribution of Target Variable
#1-Defective Heart
#0-Healthy Heart
data['target'].value_counts()

```

```
1    165
0    138
Name: target, dtype: int64
```

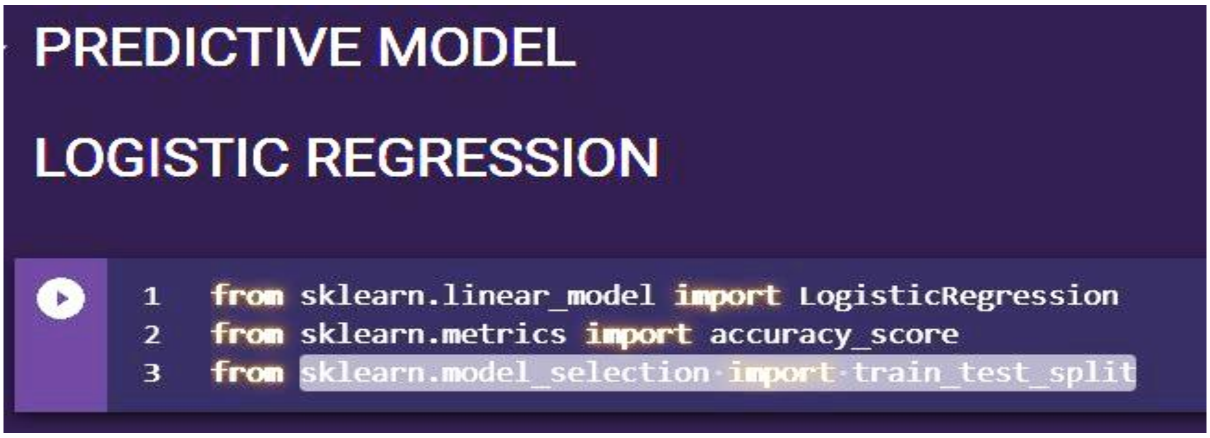


```
'''The ratio between 1 and 0 is much less than 1.5 which indicates that the target feature is not imbalanced.
 So for a balanced dataset, we can use accuracy_score as evaluation metrics for our model.'''
plt.figure(figsize=(10,8))
sns.set_context('notebook',font_scale = 1)
sns.countplot(data['target'])
plt.tight_layout()
```

- Now we used this balanced dataset to build our predictive model.
- We use Logistic Regression to build the model.
- From sklearn.linear_model we shall be importing the logistic



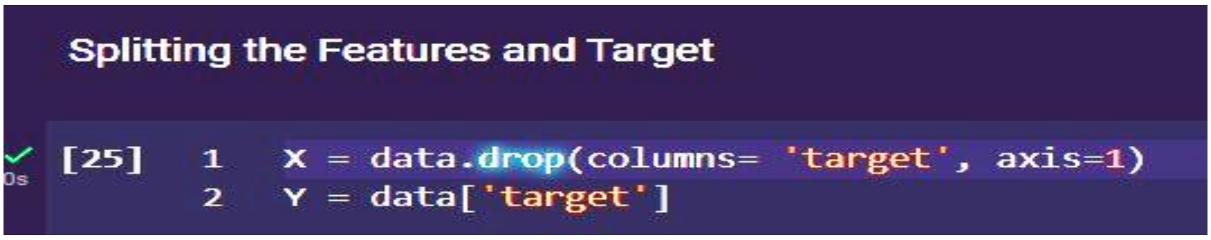
regression module and then fit training data. Let's look at the lines of code and the respective output in the form of accuracy.
- Let's import sklearn.linear_model, sklearn.metrics_accuracy, and sklearn.model_selection import train_test_split.



PREDICTIVE MODEL

LOGISTIC REGRESSION

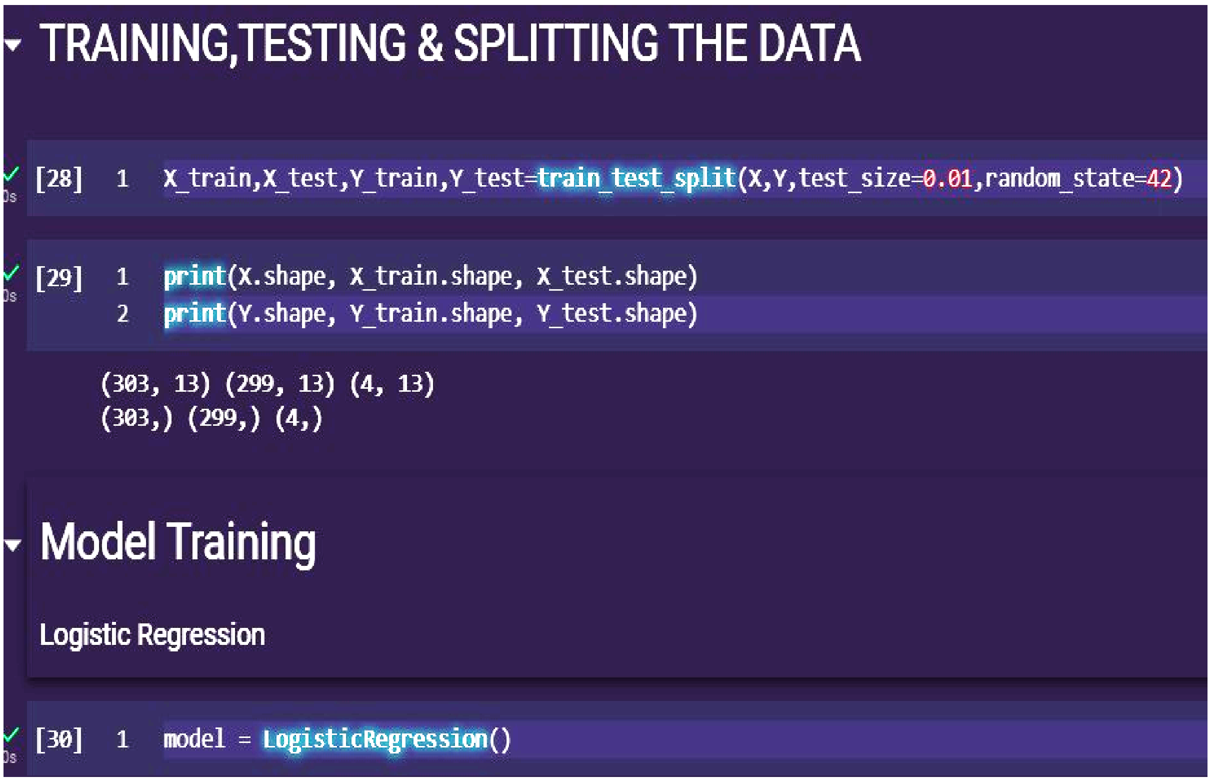
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
```



Splitting the Features and Target

```
[25]  X = data.drop(columns= 'target', axis=1)
      Y = data['target']
```

- Now splitting features and target.
- Now train and test the data and train the logistic regression model with training data.



TRAINING,TESTING & SPLITTING THE DATA

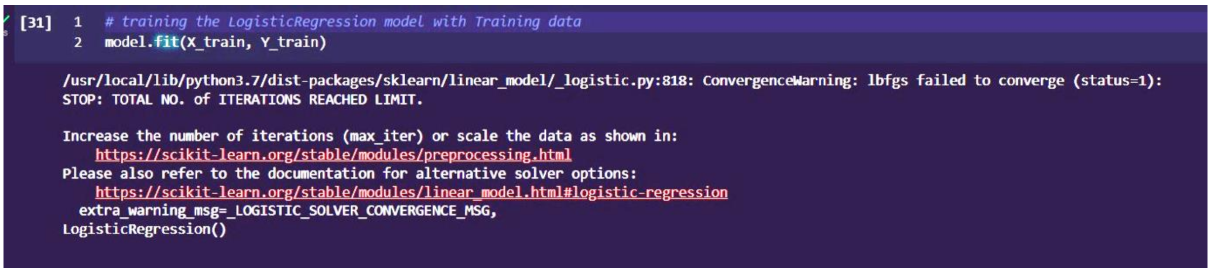
```
[28]  X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.01,random_state=42)
```

```
[29]  print(X.shape, X_train.shape, X_test.shape)
      print(Y.shape, Y_train.shape, Y_test.shape)

(303, 13) (299, 13) (4, 13)
(303,) (299,) (4,)
```

Model Training

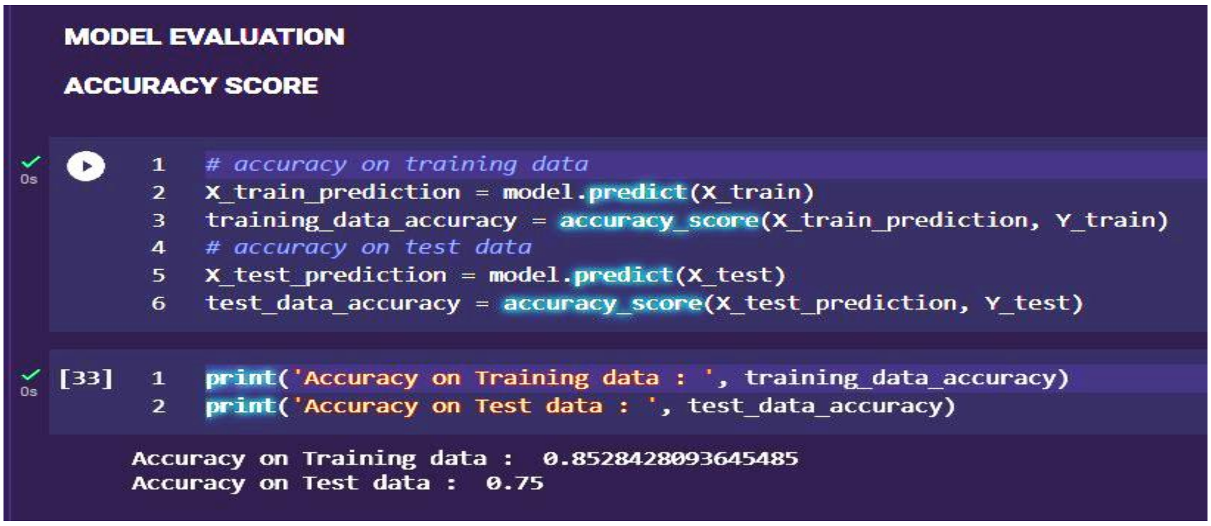
Logistic Regression

```
[30]  model = LogisticRegression()
```



```
[31]
# training the LogisticRegression model with Training data
model.fit(X_train, Y_train)

/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

- Now evaluate the model and check accuracy score.

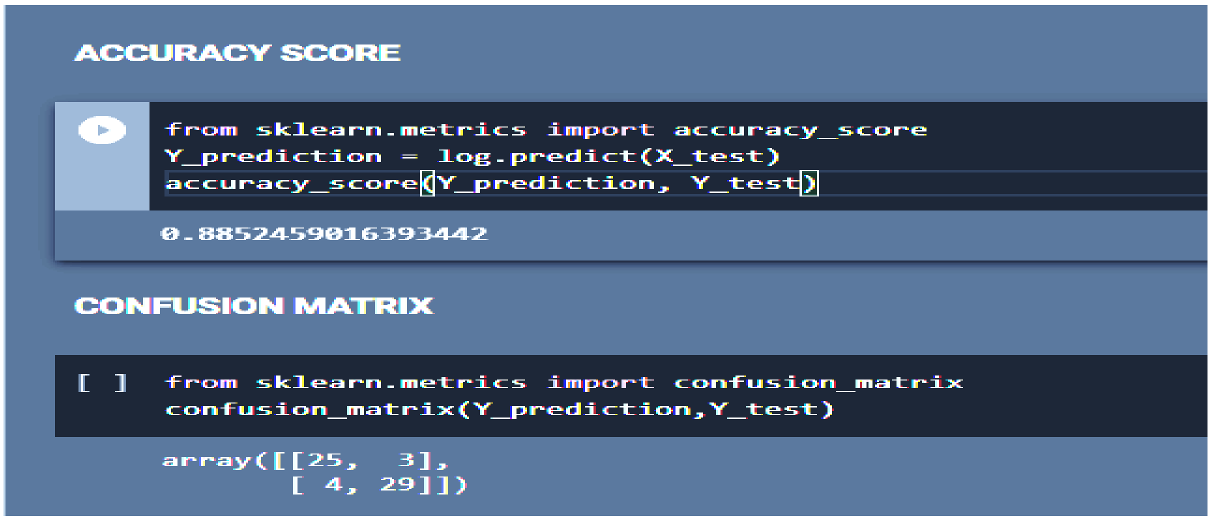


MODEL EVALUATION

ACCURACY SCORE

```
# accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)
# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

```
[33]
print('Accuracy on Training data : ', training_data_accuracy)
print('Accuracy on Test data : ', test_data_accuracy)

Accuracy on Training data :  0.8528428093645485
Accuracy on Test data :  0.75
```



ACCURACY SCORE

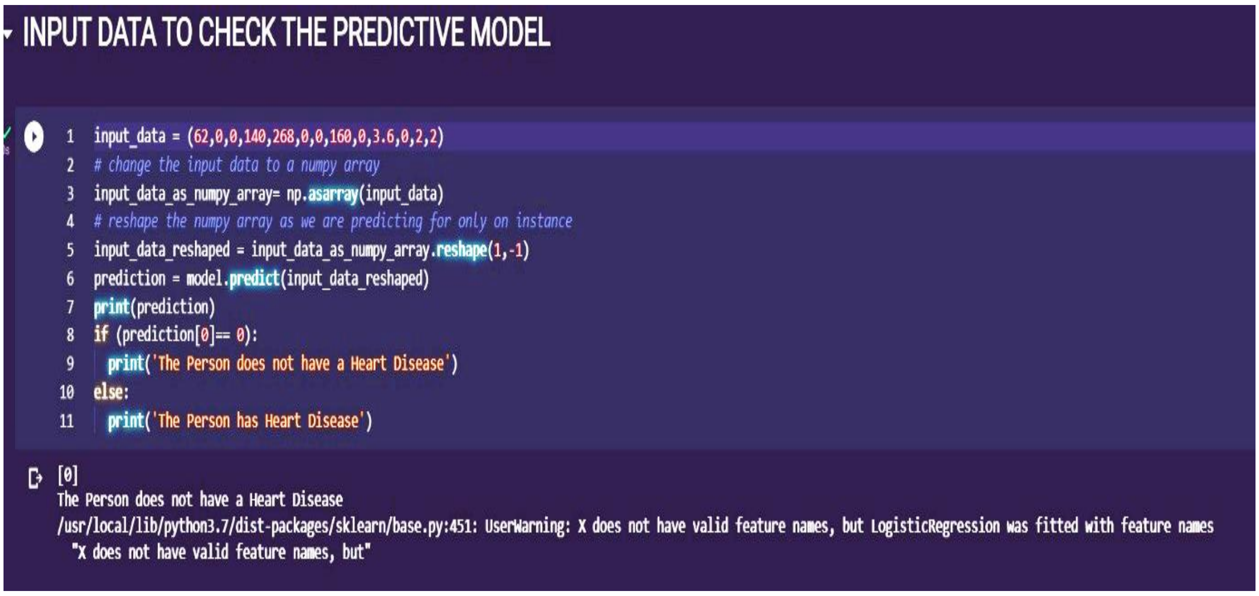
```
from sklearn.metrics import accuracy_score
Y_prediction = log.predict(X_test)
accuracy_score(Y_prediction, Y_test)

0.8852459016393442
```

CONFUSION MATRIX

```
[ ]  from sklearn.metrics import confusion_matrix
     confusion_matrix(Y_prediction,Y_test)

     array([[25,  3],
            [ 4, 29]])
```

- In Machine Learning Accuracy means the measurement used to determine which model is best at identifying relationships and patterns between variables in a dataset based on the input, or training, data. Model accuracy is defined as the number of classifications a model correctly predicts divided by the total number of predictions made. It's a way of assessing the performance of a model, but certainly not the only way. If your 'X' value is between 70% and 80%, you've got a good model. If your 'X' value is between 80% and 90%, you have an excellent model. If your 'X' value is between 90% and 100%, it's a probably an overfitting case.

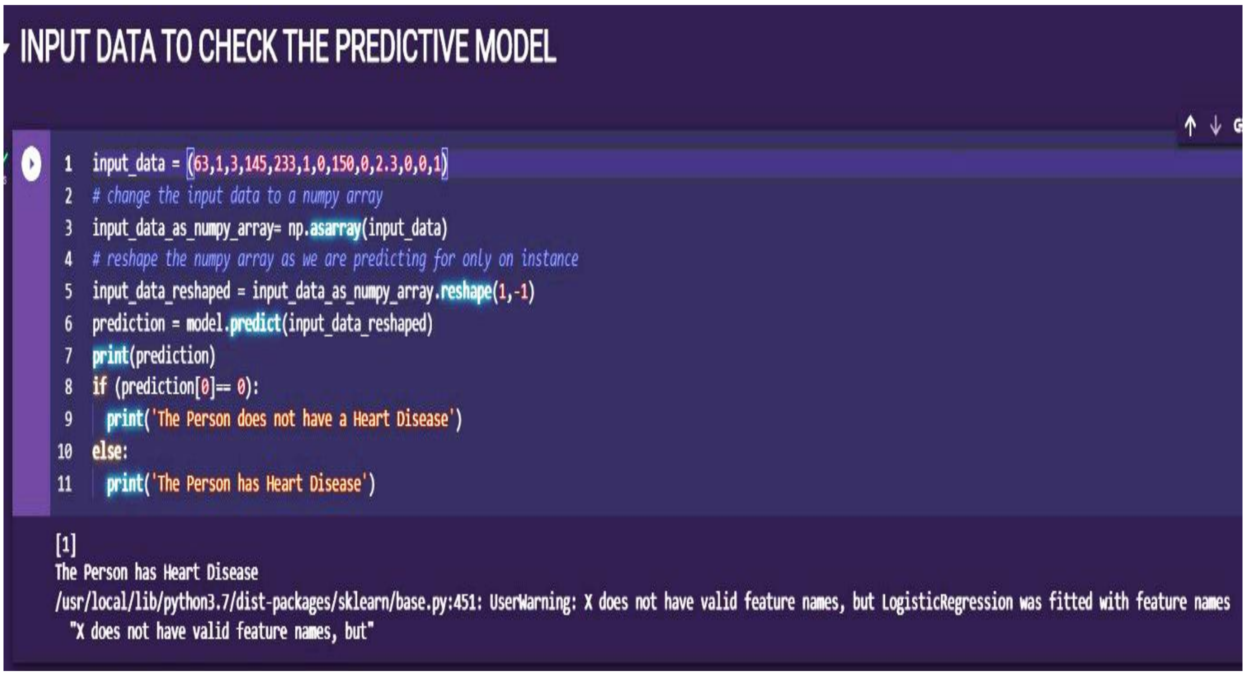
- Now Let's check the Model. Is our model perfectly predict Heart Disease or not.



INPUT DATA TO CHECK THE PREDICTIVE MODEL

```
input_data = (62,0,0,140,268,0,0,160,0,3.6,0,2,2)
# change the input data to a numpy array
input_data_as_numpy_array= np.asarray(input_data)
# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
prediction = model.predict(input_data_reshaped)
print(prediction)
if (prediction[0]== 0):
  print('The Person does not have a Heart Disease')
else:
  print('The Person has Heart Disease')
```

```
[0]
The Person does not have a Heart Disease
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but LogisticRegression was fitted with feature names
  "X does not have valid feature names, but"
```



INPUT DATA TO CHECK THE PREDICTIVE MODEL

```
input_data = (63,1,3,145,233,1,0,150,0,2.3,0,0,1)
# change the input data to a numpy array
input_data_as_numpy_array= np.asarray(input_data)
# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
prediction = model.predict(input_data_reshaped)
print(prediction)
if (prediction[0]== 0):
  print('The Person does not have a Heart Disease')
else:
  print('The Person has Heart Disease')
```

```
[1]
The Person has Heart Disease
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not have valid feature names, but LogisticRegression was fitted with feature names
  "X does not have valid feature names, but"
```

- We can see that the machine Learning model is perfectly working by input the different data.

ADVANTAGES AND DISADVANTAGES

Advantages:

1.We are using Logistic Regression in this project for Heart disease prediction which is very easy to do.

2.We have done analysis on different parameters and it’s effect on target for this anyone can understand the whole project

Disadvantages:

1.The model is not predicting 100% accuracy.

CONCLUSION

We have collected the raw data from online sources. Then we take this raw data and format it.

We did data visualization and data analysis of the target variable, age features.

Machine learning enables the detection of patterns from the data for the prediction of the patient’s condition.

FUTURE SCOPE

The data taken was limited. The project could be extended to more number of days. The weather was predicted only by taking the last three days in account, it can be predicted by looking at the last four to seven days as well.

The error can be minimized as well using other algorithms.

REFERENCES

- ❖ Artificial Intelligence- https://www.ijert.org/a-literature-survey-on-artificial-intelligence
- ○ Deep Learning- https://arxiv.org/pdf/2009.06520#:~:text=Deep%20learning%20represents%20a%20fundamental,to%20relying%20upon%20human%20intuition.
- ○ Neural Networks- https://www.ijsrp.org/research-paper-0913/ijsrp-p2148.pdf
- ○ Machine Learning- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8380165/
- • Methodology
- ❖ Machine Learning-https://www.sciencedirect.com/topics/engineering/machine-learning-method
- ○ Supervised and Unsupervised Learning- https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning#:~:text=Goals%3A%20In%20supervised%20learning%2C%20the,or%20interesting%20from%20the%20dataset.
- ○ NumPy- https://medium.com/mlpoint/numpy-for-machine-learning-211a3e58b574
- ○ SciPy- https://blog.eduonix.com/artificial-intelligence/scientific-python-scipy-machine-learning/#:~:text=Scipy%20can%20perform%20powerful%20image,dealing%20with%20Machine%20Learning%20applications.
- ○ Scikit learn- https://scikit-learn.org/
- ○ Pandas- https://medium.com/mlpoint/pandas-for-machine-learning-53846bc9a98b
- ○ Regression Analysis- https://www.javatpoint.com/regression-analysis-in-machine-learning
- ○ Matplotlib- https://www.javatpoint.com/matplotlib
- ○ Clustering- https://www.javatpoint.com/clustering-in-machine-learning
- • HEART DISEASE PREDICTION- https://scholar.google.co.in/scholar?q=heart+disease+prediction+using+machine+learning&hl=en&as_sdt=0&as_vis=1&oi=scholart
- ❖ Results And Discussions- https://colab.research.google.com/drive/1G_vAcCp_bJ1s079x1d5zMp0jWEkrGE2i
- ❖ Dataset- https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset

Thank You