A Data-science Report on:

“Data science in healthcare and Medical imaging”

Prepared by : _SANDEEP RATHOD

Roll. No. : _U19CS022

Class : B.Tech –III (Computer Science & Engineering) 5th Semester

Year : 2021-22

Guided by : Shivangi Modi

Department of Computer Engineering
Sardar Vallabhbhai National Institute of Technology,
Surat -395007 (Gujarat), India

Rathod

U19CS022
(SANDEEP RATHOD)

JUST A SMALL GUIDE TO THE REPORT.

In 1st chapter:

I have practically implemented data analysis on a data set on heart patients to show the datacycles and all the machine learning algo's to predict the Hearth attack on the patients.

(And got the result as 88.5% accuracy.)

1.Used data set from Kaggle website

2.jupiter notebook (App for running the python code and machine learning algorithms easily).

3.all screenshots are added in the report.

In 2nd chapter:

Was not comfortable with the 2d data so didn't implemented practically but theoretical explaination is given in here.

Contents

List of Figures

Data science in Health care

Data Science helps in advancing healthcare facilities and processes. It helps boost productivity in diagnosis and treatment and enhance the workflow of healthcare systems. The ultimate goals of the healthcare system are as follows:

To ease the workflow of the healthcare system
To reduce the risk of treatment failure
To provide proper treatment on time
To avoid unnecessary emergency due to the non-availability of doctors
To reduce the waiting time of patients

Data science applications In healthcare

1.Data Management and Data Governance
2.Workflow Optimization and Process Improvements
3.Medical Image Analysis
4.Genetics/ Genomics - Treatment personalization
5.Predictive Analytics and Healthcare

Medicine and healthcare are two of the most crucial aspects of our life as humans. Medicine has traditionally relied completely on the discretion advised by doctors. A doctor, for example, would have to recommend appropriate therapies depending on a patient's symptoms. This wasn't always the case, and it was prone to human mistake. It is now possible to gather precise diagnostic measures thanks to advances in computers and, in particular, Data Science. Data science is used in a variety of sectors in healthcare, including medical imaging, drug development, genomics, predictive diagnosis, and many others. We'll go over each field with examples one by one.

Chapter 1

Data science Life cycle data analysis on heart patients and predicting the stroke attack

1.1 Introduction

The data life cycle, also known as the information life cycle, relates to the duration of data storage in your system. This life cycle describes all of the stages that your data goes through from the time it is first captured to the time it is destroyed. It includes like data collection and data preparation and data analysis and visualization. and others in this chapter we are going analyse the data of a heart patient and try to build the machine learning model which predicts the stroke attach. with following all the data life cycle processes.

1.2 Practical implementation and showing the data cycle on heart petient dataset

For the for the analysis
we have taken data from kaggle website of heart certain patient information and we are going to analysis the data in jupiter notebook using python 3.

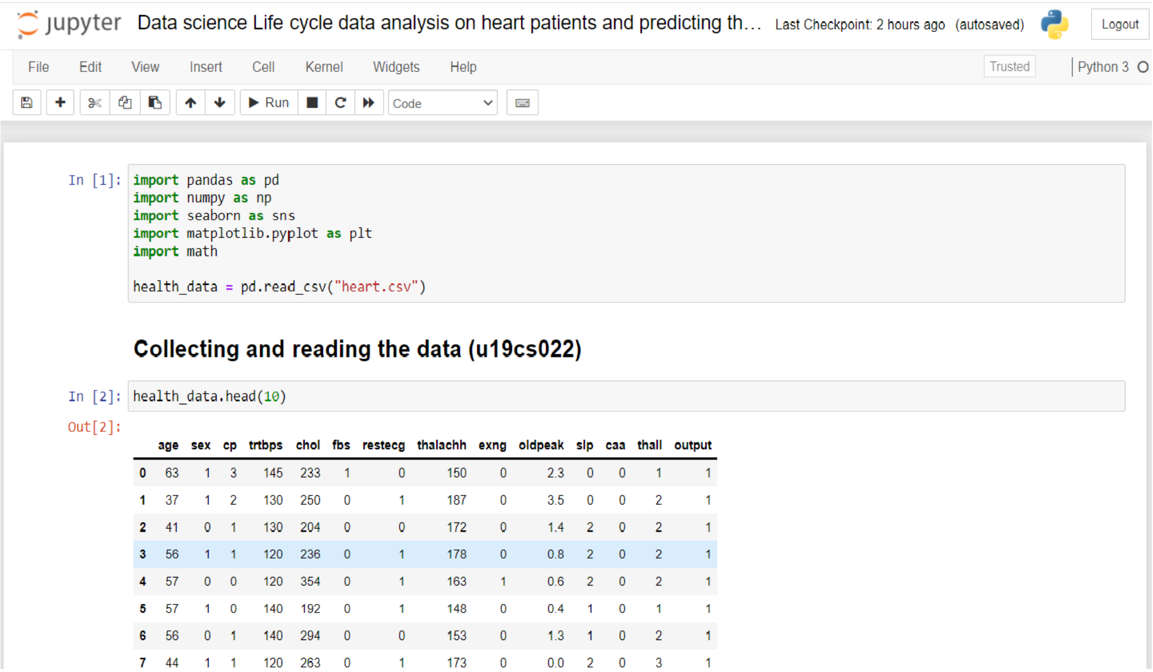
1.2.1 Data collection and reading the data

Here the data is collected from kaggle website which is the data of certain heart patients.the data looks like this. Where here he data has the 14 columns and the the column names are as in the image

before that i have imported some of the necessory files as pandas , numpy and seaborn etc...

now lets move to the next step of data cycle

!h


jupyter
Data science Life cycle data analysis on heart patients and predicting th…
Last Checkpoint: 2 hours ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

```
In [1]: import pandas as pd
        import numpy as np
        import seaborn as sns
        import matplotlib.pyplot as plt
        import math

        health_data = pd.read_csv("heart.csv")
```

Collecting and reading the data (u19cs022)

```
In [2]: health_data.head(10)
```

Out[2]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
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

Figure 1.1: Hearth patients data

1.3 Data Preperation

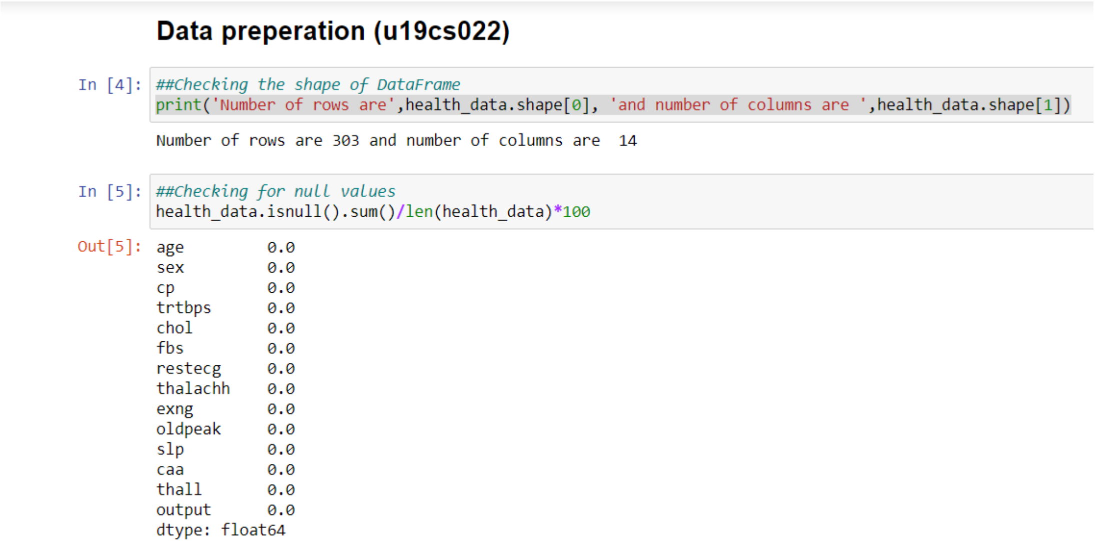
A data scientist must first examine the data to identify any gaps or data that do not add any value. During this process, you must go through several steps, including:
• Data Integration: Resolve any conflicts in the dataset and eliminate redundancies
• Data Transformation: Normalize, transform and aggregate data using ETL (extract, transform, load) methods • Data Reduction: Using various strategies, reduce the size of data without impacting the quality or outcome
• Data Cleaning: Correct inconsistent data by filling out missing values and smoothing out noisy data

Lets see this data preperation method added in our project..

For that a bit data analysis is required as to know the attributes of the column and the datatype of the columps and the number of null values and then all.

1.3.1 data analysis

Here we are going to check the shape of the data frame..
and checking the null values
Checking For datatypes of the attributes(u19cs022)
Cheaking for duplicate rows



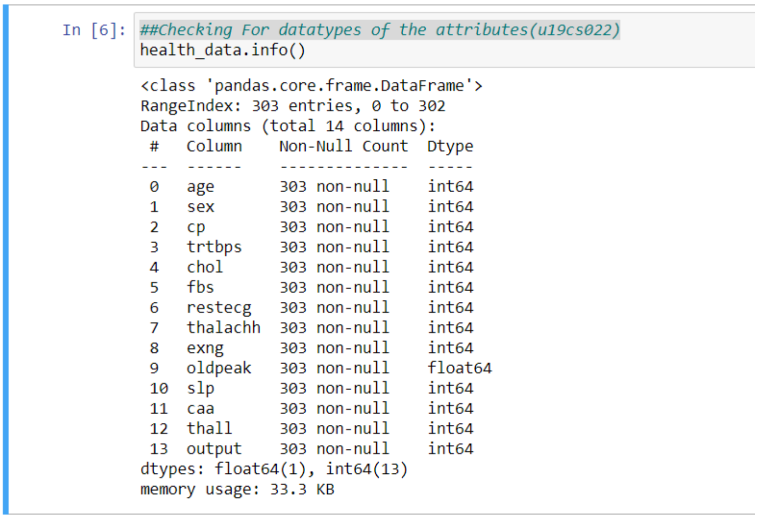
Data preperation (u19cs022)

```
In [4]: ##Checking the shape of DataFrame
        print('Number of rows are',health_data.shape[0], 'and number of columns are ',health_data.shape[1])

        Number of rows are 303 and number of columns are  14
```

```
In [5]: ##Checking for null values
        health_data.isnull().sum()/len(health_data)*100

Out[5]: age         0.0
        sex         0.0
        cp          0.0
        trtbps      0.0
        chol        0.0
        fbs         0.0
        restecg     0.0
        thalachh    0.0
        exng        0.0
        oldpeak     0.0
        slp         0.0
        caa         0.0
        thall       0.0
        output      0.0
        dtype: float64
```

!h

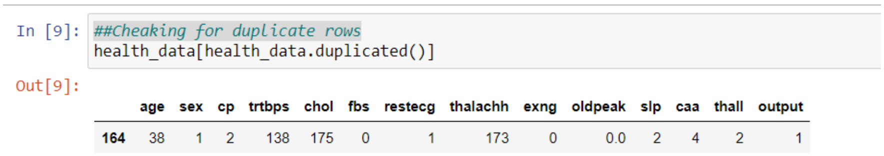
Figure 1.2: Checking the null values



```
In [6]: ##Checking For datatypes of the attributes(u19cs022)
        health_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   trtbps    303 non-null    int64  
         4   chol      303 non-null    int64  
         5   fbs       303 non-null    int64  
         6   restecg   303 non-null    int64  
         7   thalachh  303 non-null    int64  
         8   exng      303 non-null    int64  
         9   oldpeak   303 non-null    float64
         10  slp       303 non-null    int64  
         11  caa       303 non-null    int64  
         12  thall     303 non-null    int64  
         13  output    303 non-null    int64  
        dtypes: float64(1), int64(13)
        memory usage: 33.3 KB
```

!h

Figure 1.3: Checking the datatype of all variables



```
In [9]: ##Cheaking for duplicate rows
        health_data[health_data.duplicated()]

Out[9]:
```

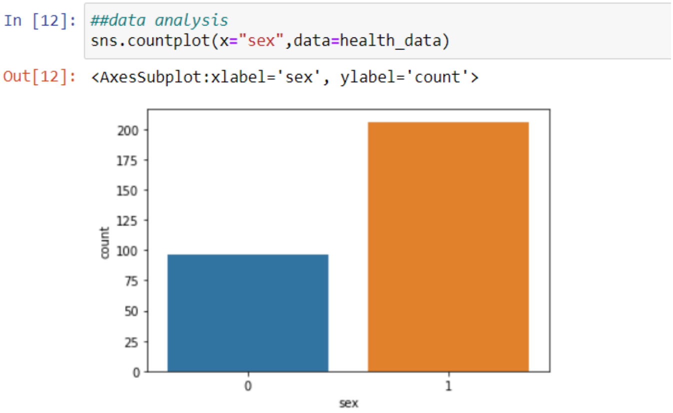
	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
164	38	1	2	138	175	0	1	173	0	0.0	2	4	2	1

!h

Figure 1.4: Checking for duplicates

1.3.2 data analysis by ploting the graph to show the relationship between the different variables or columns

1. graph on how many males have got the stroke the compare to the females.

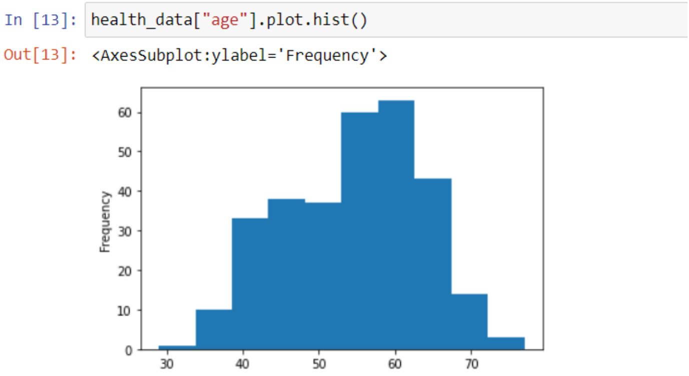


```
In [12]: ##data analysis
         sns.countplot(x="sex",data=health_data)
Out[12]: <AxesSubplot:xlabel='sex', ylabel='count'>
```

Figure 1.5: Male and Female (Comparison)

here the female number is looking because as of getting stroke so in the collected data females numbers are double than the males...

2. A histogram graph to show the age group of getting stroke

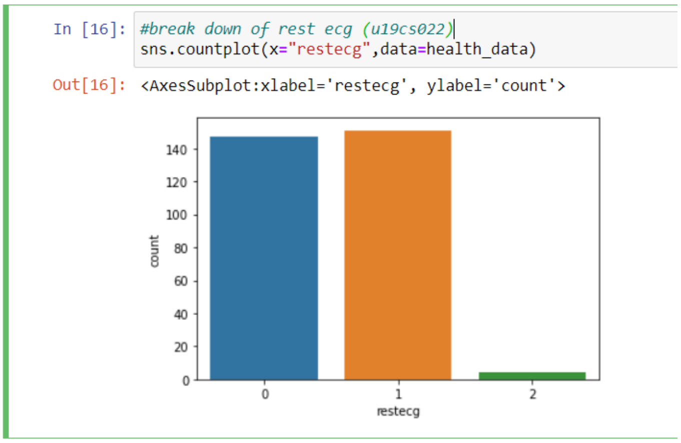


```
In [13]: health_data["age"].plot.hist()
Out[13]: <AxesSubplot:ylabel='Frequency'>
```

Figure 1.6: Age historigraph of patients

From the image you can conclude that the age between 55 and 65 has the higher chance of getting stroke.

3. Breaking down the ecg data of the patients
here the
1 151
0 147
2 4
Name: restecg, dtype: int64



```
In [16]: #break down of rest ecg (u19cs022)
         sns.countplot(x="restecg",data=health_data)

Out[16]: <AxesSubplot:xlabel='restecg', ylabel='count'>
```

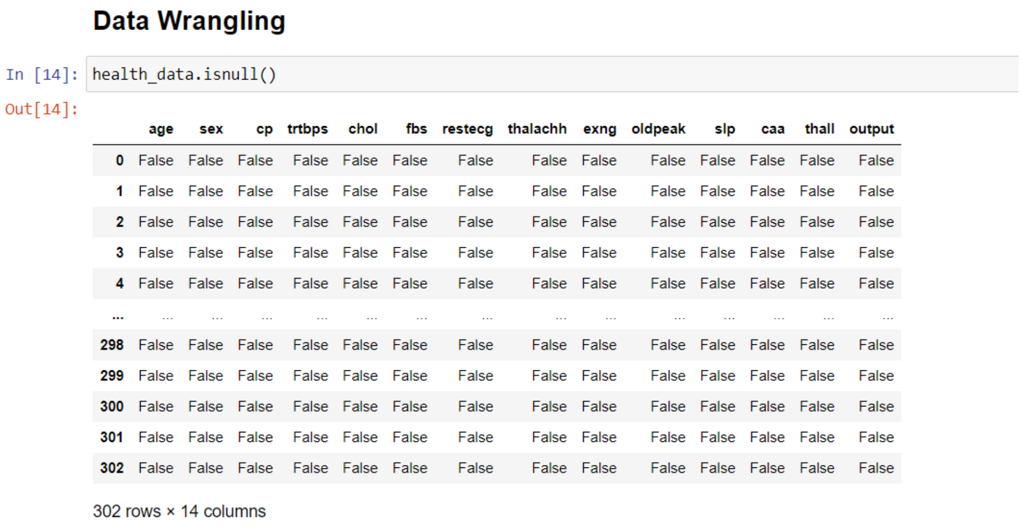
Figure 1.7: Rest ECG graph of patient

1.4 Data Wrangling

CLEAN THE DATA BY REMOVING THE NAN VALUES AND UNNECESARY COLUMNS INTHE DATASET.

-

Checking the null values



Data Wrangling

```
In [14]: health_data.isnull()

Out[14]:
```

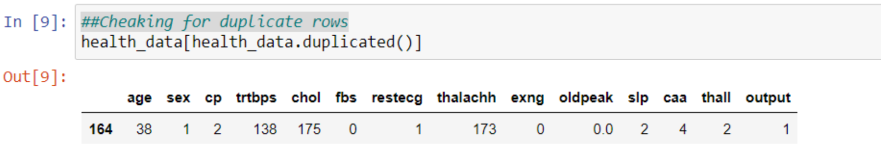
	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	False	False	False	False	False	False	False	False	False	False	False	False	False	False
299	False	False	False	False	False	False	False	False	False	False	False	False	False	False
300	False	False	False	False	False	False	False	False	False	False	False	False	False	False
301	False	False	False	False	False	False	False	False	False	False	False	False	False	False
302	False	False	False	False	False	False	False	False	False	False	False	False	False	False

302 rows × 14 columns

Figure 1.8: Finding the null values

from the above information we can see there is no element where there is null value so we cannot clean the data here.

Now checking any duplicate values are there in the dataset.

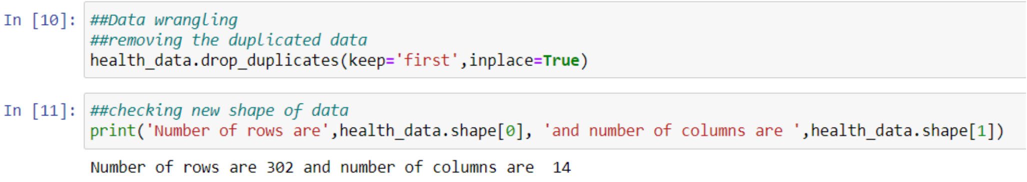


```
In [9]:  ##Cheaking for duplicate rows
          health_data[health_data.duplicated()]

Out[9]:
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
164	38	1	2	138	175	0	1	173	0	0.0	2	4	2	1

Figure 1.9: Finding the duplicate value

Checking from the above image there is a duplicate value which we have to remove or you can say cleaning the data.



```
In [10]: ##Data wrangling
         ##removing the duplicated data
         health_data.drop_duplicates(keep='first',inplace=True)

In [11]: ##checking new shape of data
         print('Number of rows are',health_data.shape[0], 'and number of columns are ',health_data.shape[1])

         Number of rows are 302 and number of columns are  14
```

Figure 1.10: Checking the data shape after data cleaning

With datacleaning we removed the duplicated data and checked the new shaped dataset.

For the other columns like for sex we could have changed from female to give some numerical values for the ease of operation but all those things are already done so no need to some extra unneccesary cleaning process.

so from here Data wrangling completes.

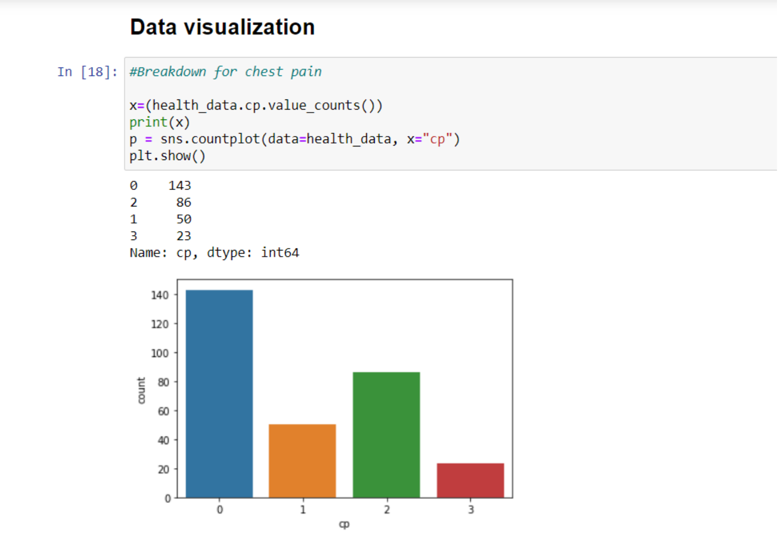
1.5 Data Visualization

Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

In the world of Big Data, data visualization tools and technologies are essential to analyze massive amounts of information and make data-driven decisions.

Breakdown for chest pain

Checking for how many people got the stroke who where havign the certain value of chest pain.//

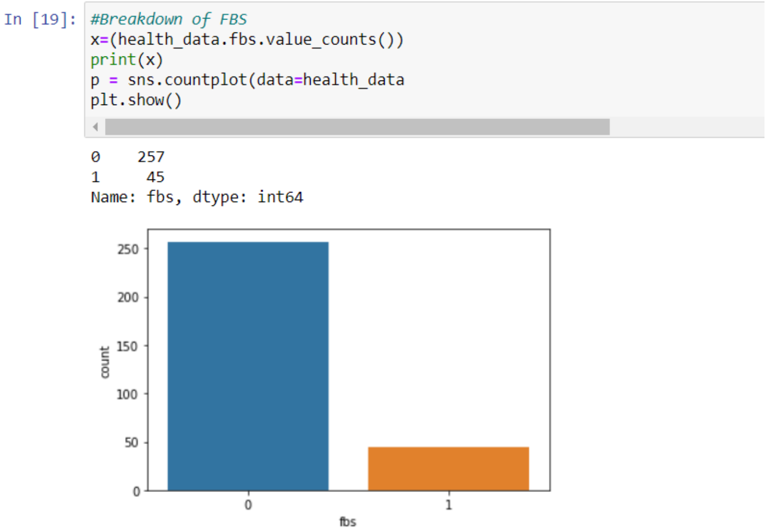


Data visualization

```
In [18]: #Breakdown for chest pain

         x=(health_data.cp.value_counts())
         print(x)
         p = sns.countplot(data=health_data, x="cp")
         plt.show()
```

```
0    143
2     86
1     50
3     23
Name: cp, dtype: int64
```

Figure 1.11: On Chest pain



```
In [19]: #Breakdown of FBS
         x=(health_data.fbs.value_counts())
         print(x)
         p = sns.countplot(data=health_data
         plt.show()
```

```
0    257
1     45
Name: fbs, dtype: int64
```

Figure 1.12: On FBS

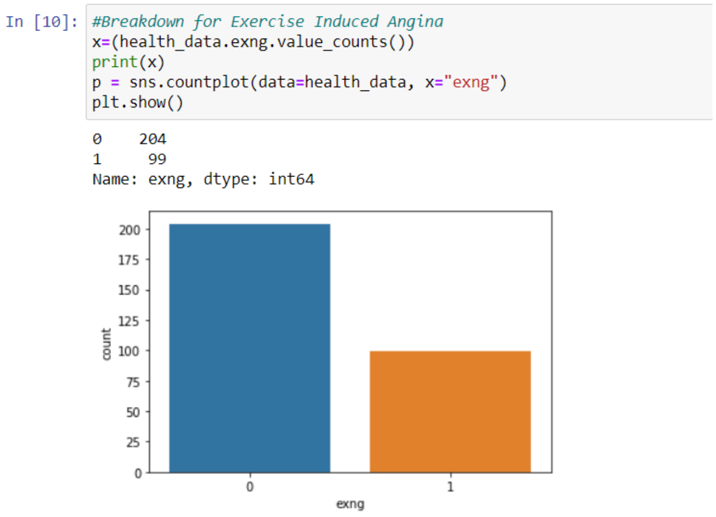
With the graphs we can observe that.

1.It can be observed people have chest pain of type 0 i.e 'Typical Angina' is the highest.

2.It can be observed people have chest pain of type 3 i.e 'Asymptomatic' is the lowest

3.It can also be observed people with chest pain of type 0 is almost 50
these are the conclusion made with the above graph which looks quite generes now lets continue the visualizing the data with other vaiarbles.

Breakdown for Exercise Induced Angina



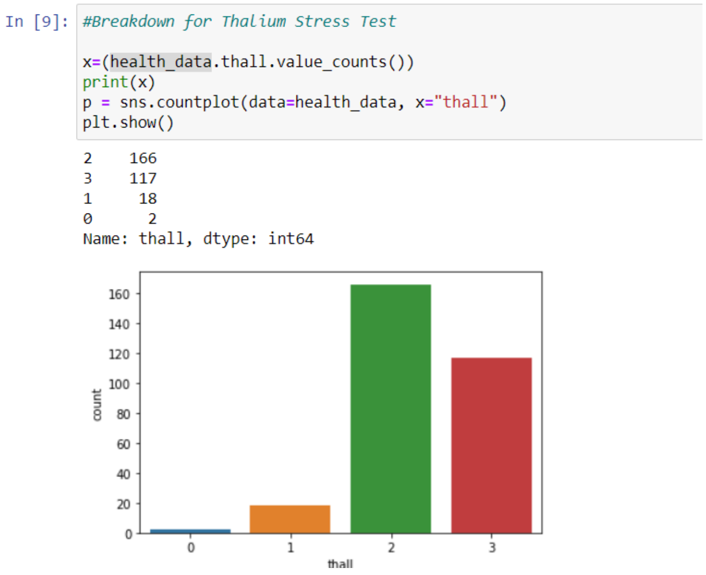
```
In [10]: #Breakdown for Exercise Induced Angina
         x=(health_data.exng.value_counts())
         print(x)
         p = sns.countplot(data=health_data, x="exng")
         plt.show()
```

```
0    204
1     99
Name: exng, dtype: int64
```

Figure 1.13: Exercise Induced Angina

FBS with value 0 is significantly higher than value 1.

Thalium stress test



```
In [9]: #Breakdown for Thalium Stress Test

        x=(health_data.thall.value_counts())
        print(x)
        p = sns.countplot(data=health_data, x="thall")
        plt.show()
```

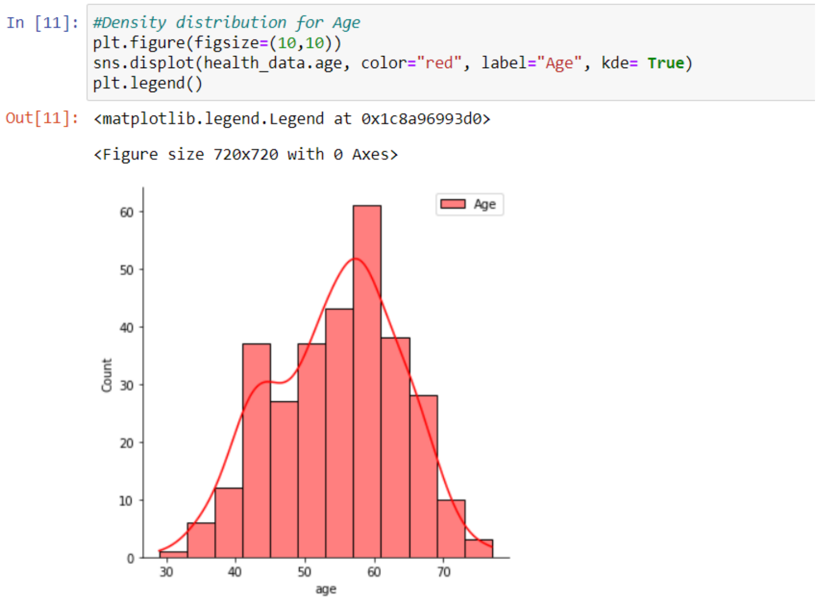
```
2    166
3    117
1     18
0      2
Name: thall, dtype: int64
```

Figure 1.14: Thalium stress tes

With observing the above graph..

EXNG count is more than double for type 0

Density distribution for Age

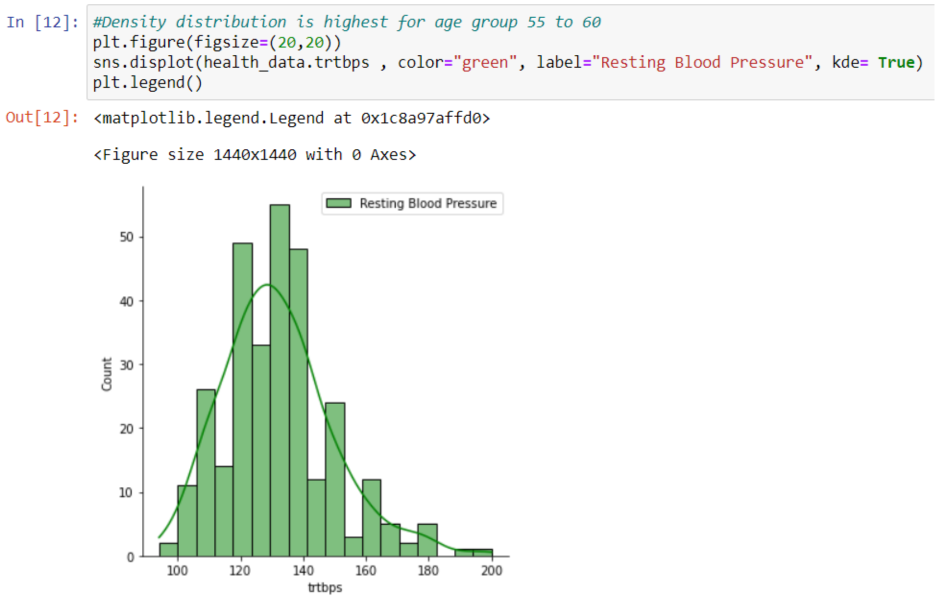


```
In [11]: #Density distribution for Age
         plt.figure(figsize=(10,10))
         sns.displot(health_data.age, color="red", label="Age", kde= True)
         plt.legend()

Out[11]: <matplotlib.legend.Legend at 0x1c8a96993d0>

         <Figure size 720x720 with 0 Axes>
```

Figure 1.15: density distribution of age

Density distribution is highest for age group 55 to 60.

Resting blood pressure

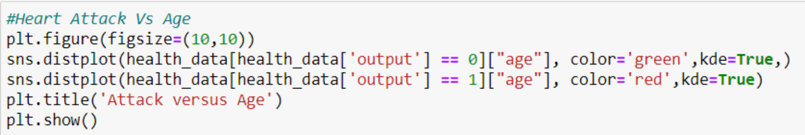


```
In [12]: #Density distribution is highest for age group 55 to 60
         plt.figure(figsize=(20,20))
         sns.displot(health_data.trtbps , color="green", label="Resting Blood Pressure", kde= True)
         plt.legend()

Out[12]: <matplotlib.legend.Legend at 0x1c8a97affd0>

         <Figure size 1440x1440 with 0 Axes>
```

Figure 1.16: Blood pressure Vs heart attack

the heart attack is more for the ones who blood pressure in above 130.

Heart Attack Vs Age



```
#Heart Attack Vs Age
plt.figure(figsize=(10,10))
sns.distplot(health_data[health_data['output'] == 0]["age"], color='green',kde=True,)
sns.distplot(health_data[health_data['output'] == 1]["age"], color='red',kde=True)
plt.title('Attack versus Age')
plt.show()
```

Figure 1.17: Code

Checking all graphs of different parameters with the age.

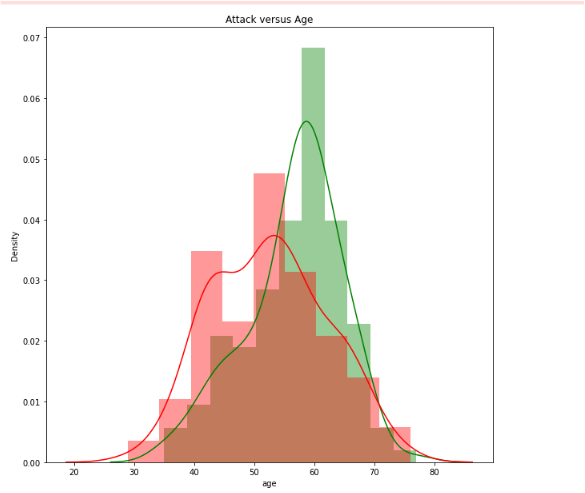


Figure 1.18: Heart attack Vs Age

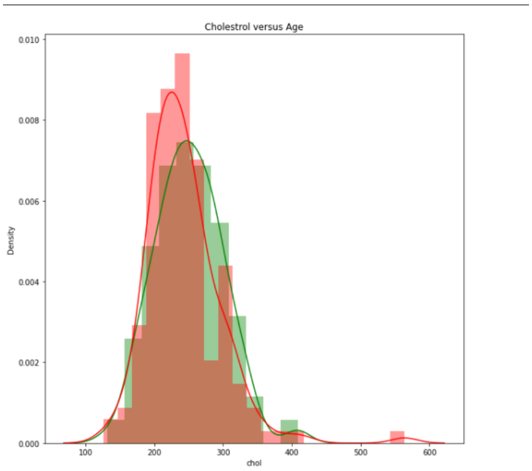


Figure 1.19: Cholestrol Vs Age

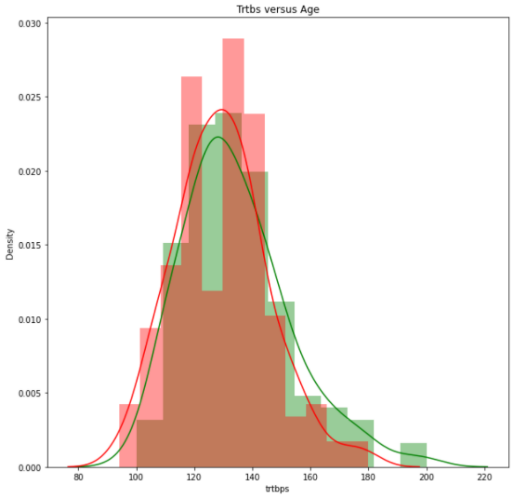


Figure 1.20: Trtbs Vs Age

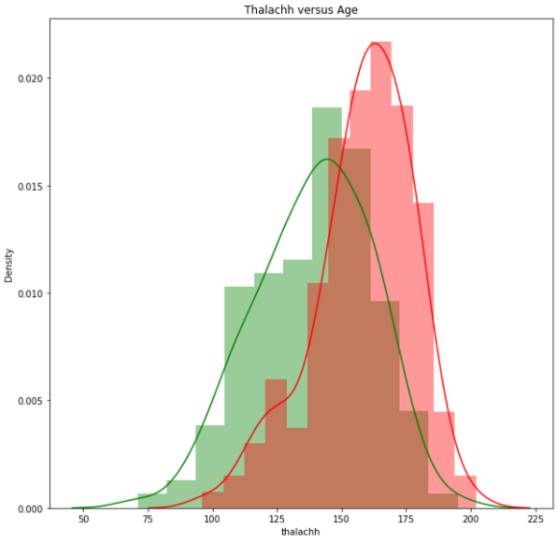


Figure 1.21: Thalachh Vs Age

By ploting all the graphs with different parameters with age so that we can analyse which disease are more common at which age which are cause most for the heart attack.

we got to see that the most vulnerable age for the heart attack is 50 to 60.

with this the data visualization is complete from now on we are going to do Data preprocessing.

1.6 Data Preprocessing

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task. Lets do Data preprocessing one by one.

Step 1

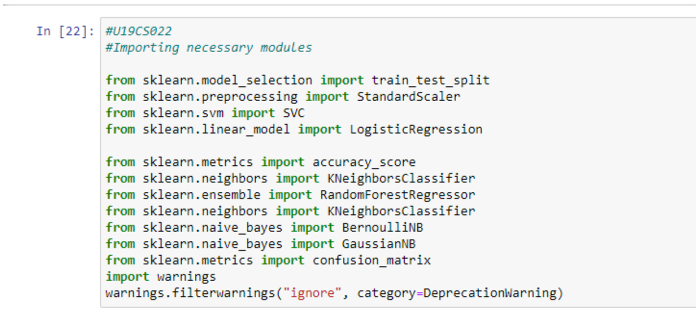
There's no need for categorical encoding..

as all data are in numerical format so we don't have to encode for categorical data..

Step 2

Splitting the dataset into training and testing data

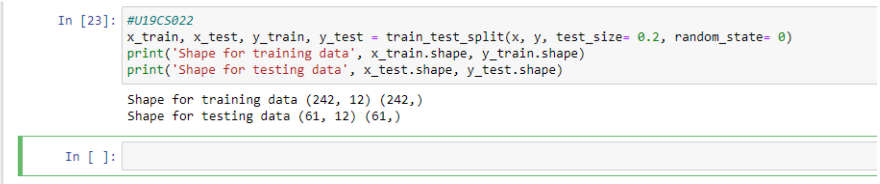
Importing necessary modules.



```
In [22]: #U19CS022
         #Importing necessary modules

         from sklearn.model_selection import train_test_split
         from sklearn.preprocessing import StandardScaler
         from sklearn.svm import SVC
         from sklearn.linear_model import LogisticRegression

         from sklearn.metrics import accuracy_score
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.ensemble import RandomForestRegressor
         from sklearn.neighbors import KNeighborsClassifier
         from sklearn.naive_bayes import BernoulliNB
         from sklearn.naive_bayes import GaussianNB
         from sklearn.metrics import confusion_matrix
         import warnings
         warnings.filterwarnings("ignore", category=DeprecationWarning)
```

Figure 1.22: Importing necessary modules

The splitted data showing..

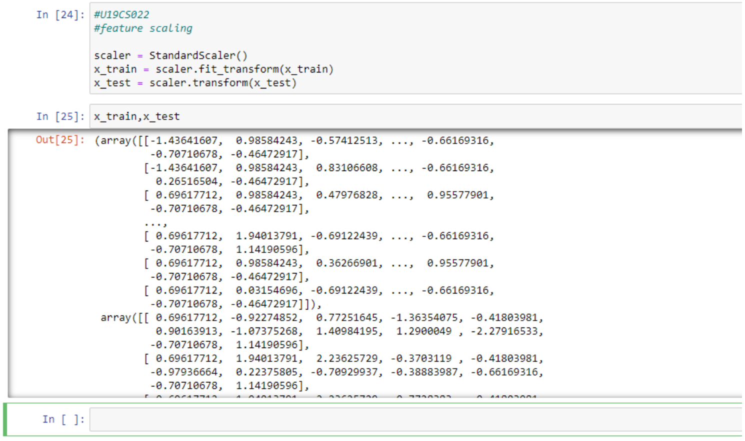


```
In [23]: #U19CS022
         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size= 0.2, random_state= 0)
         print('Shape for training data', x_train.shape, y_train.shape)
         print('Shape for testing data', x_test.shape, y_test.shape)

         Shape for training data (242, 12) (242,)
         Shape for testing data (61, 12) (61,)

In [ ]:
```

Figure 1.23: Data splitting in test and train data

Step 3

feature scaling



```
In [24]: #U19CS022
         #feature scaling

         scaler = StandardScaler()
         x_train = scaler.fit_transform(x_train)
         x_test = scaler.transform(x_test)

In [25]: x_train,x_test

Out[25]: (array([[-1.43641607,  0.98584243, -0.57412513, ..., -0.66169316,
                  -0.70710678, -0.46472917],
                 [-1.43641607,  0.98584243,  0.83106608, ..., -0.66169316,
                   0.26516504, -0.46472917],
                 [ 0.69617712,  0.98584243,  0.47976828, ...,  0.95577901,
                  -0.70710678, -0.46472917],
                 ...,
                 [ 0.69617712,  1.94013791, -0.69122439, ..., -0.66169316,
                  -0.70710678,  1.14190596],
                 [ 0.69617712,  0.98584243,  0.36266901, ...,  0.95577901,
                  -0.70710678, -0.46472917],
                 [ 0.69617712,  0.03154696, -0.69122439, ..., -0.66169316,
                  -0.70710678, -0.46472917]]),
          array([[ 0.69617712, -0.92274852,  0.77251645, -1.36354075, -0.41803981,
                   0.90163913, -1.07375268,  1.40984195,  1.2900049 , -2.27916533,
                  -0.70710678,  1.14190596],
                 [ 0.69617712,  1.94013791,  2.23625729, -0.3703119 , -0.41803981,
                  -0.97936664,  0.22375805, -0.70929937, -0.38883987, -0.66169316,
                  -0.70710678,  1.14190596],

In [ ]:
```

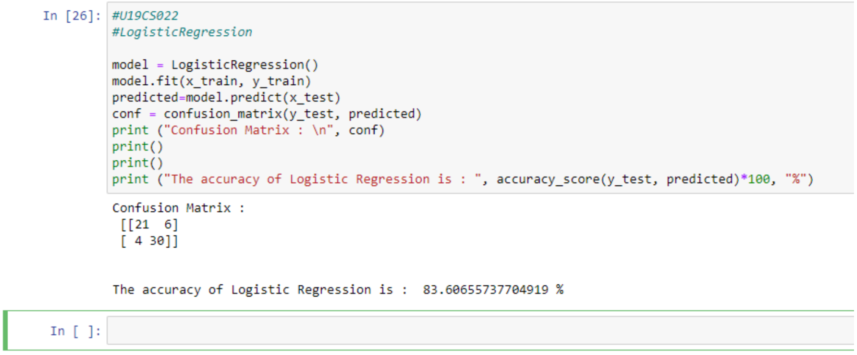
Figure 1.24: feature scaling

1.7 USING ALL MACHINE LEARNING ALGO'S TO PREDICT THE DATA

As we are all set as tested and trained the data.

Its the time to use the different machine learning algorithms to predicts the out come to complete our assignment for predicting the out come weather a patient can have hearth attack or not.

1.7.1 1. Logistic Regression

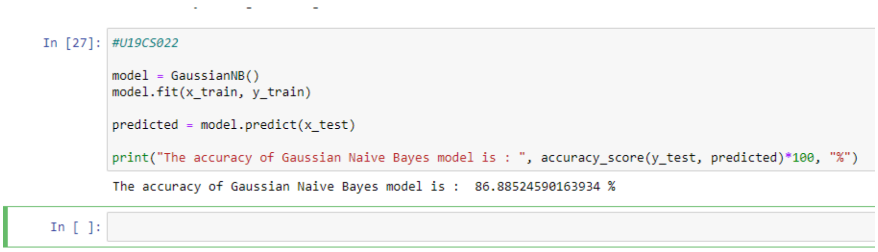


```
In [26]: #U19CS022
         #LogisticRegression

         model = LogisticRegression()
         model.fit(x_train, y_train)
         predicted=model.predict(x_test)
         conf = confusion_matrix(y_test, predicted)
         print ("Confusion Matrix : \n", conf)
         print()
         print()
         print ("The accuracy of Logistic Regression is : ", accuracy_score(y_test, predicted)*100, "%")

         Confusion Matrix :
          [[21  6]
          [ 4 30]]


         The accuracy of Logistic Regression is :  83.60655737704919 %

In [ ]:
```

Figure 1.25: Logistic Regression

The accuracy of Logistic Regression is : 83.60655737704919
This accuracy is very good. lets check with ather algorithms.

1.7.2 2. Gaussian Naive Bayes

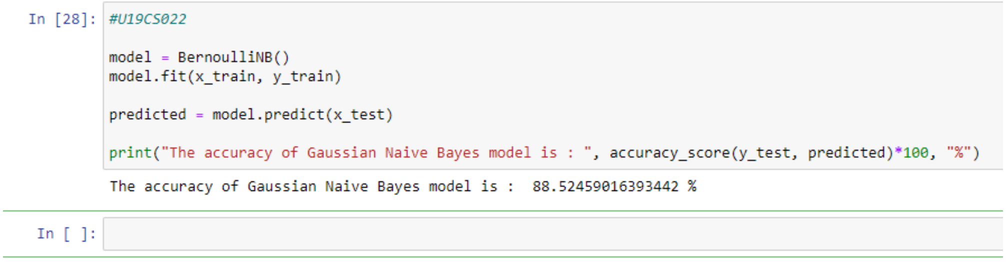


```
In [27]: #U19CS022

         model = GaussianNB()
         model.fit(x_train, y_train)

         predicted = model.predict(x_test)

         print("The accuracy of Gaussian Naive Bayes model is : ", accuracy_score(y_test, predicted)*100, "%")

         The accuracy of Gaussian Naive Bayes model is :  86.88524590163934 %

In [ ]:
```

Figure 1.26: Gaussian Naive Bayes

The accuracy of Gaussian Naive Bayes model is : 86.88524590163934 percentage

1.7.3 3.Bernoulli Naive Bayes

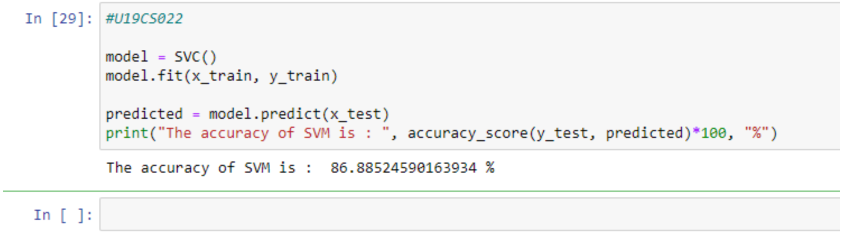


```
In [28]: #U19CS022

         model = BernoulliNB()
         model.fit(x_train, y_train)

         predicted = model.predict(x_test)

         print("The accuracy of Gaussian Naive Bayes model is : ", accuracy_score(y_test, predicted)*100, "%")

         The accuracy of Gaussian Naive Bayes model is :  88.52459016393442 %

In [ ]:
```

Figure 1.27: Bernoulli Naive Bayes

The accuracy of Gaussian Naive Bayes model is : 88.52459016393442 percentage
True Positive + True Negative : 54
False Positive + False Negative : 7

1.7.4 4. Support Vector Machine

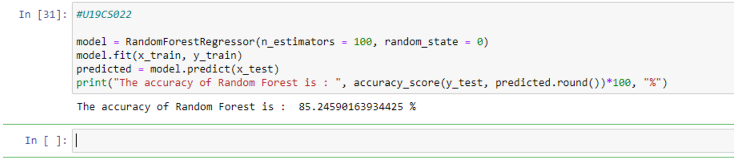


```
In [29]: #U19CS022

         model = SVC()
         model.fit(x_train, y_train)

         predicted = model.predict(x_test)
         print("The accuracy of SVM is : ", accuracy_score(y_test, predicted)*100, "%")

         The accuracy of SVM is :  86.88524590163934 %

In [ ]:
```

Figure 1.28: Support Vector Machine

The accuracy of SVM is : 86.88524590163934 percentage

1.7.5 5. Random Forest

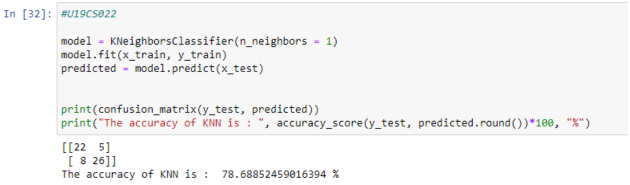


```
In [31]: #U19CS022

         model = RandomForestRegressor(n_estimators = 100, random_state = 0)
         model.fit(x_train, y_train)
         predicted = model.predict(x_test)
         print("The accuracy of Random Forest is : ", accuracy_score(y_test, predicted.round())*100, "%")

         The accuracy of Random Forest is :  85.24590163934425 %

In [ ]: 
```

Figure 1.29: Random Forest

The accuracy of Random Forest is : 85.24590163934425 percentage

1.7.6 6. K Nearest Neighbours

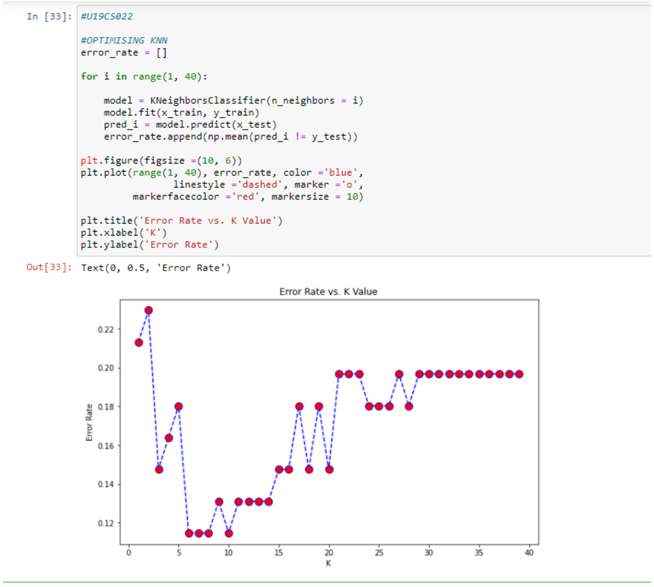


```
In [32]: #U19CS022

         model = KNeighborsClassifier(n_neighbors = 1)
         model.fit(x_train, y_train)
         predicted = model.predict(x_test)


         print(confusion_matrix(y_test, predicted))
         print("The accuracy of KNN is : ", accuracy_score(y_test, predicted.round())*100, "%")

         [[22  5]
          [ 8 26]]
         The accuracy of KNN is :  78.68852459016394 %
```

Figure 1.30: K Nearest Neighbours

The accuracy of KNN is : 78.68852459016394 percentage

Optimizing the KNN

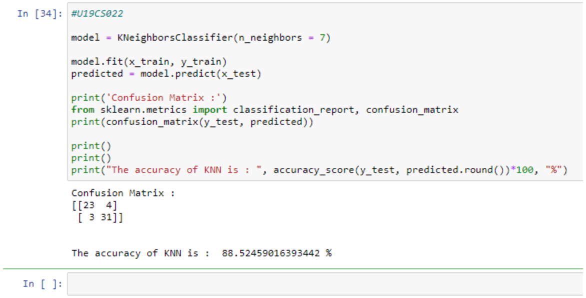


```
In [33]: #U19CS022

         #OPTIMISING KNN
         error_rate = []

         for i in range(1, 40):

             model = KNeighborsClassifier(n_neighbors = i)
             model.fit(x_train, y_train)
             pred_i = model.predict(x_test)
             error_rate.append(np.mean(pred_i != y_test))

         plt.figure(figsize =(10, 6))
         plt.plot(range(1, 40), error_rate, color ='blue',
                         linestyle ='dashed', marker ='o',
                 markerfacecolor ='red', markersize = 10)

         plt.title('Error Rate vs. K Value')
         plt.xlabel('K')
         plt.ylabel('Error Rate')

Out[33]: Text(0, 0.5, 'Error Rate')
```

Figure 1.31: Finding the errors

With k=7 as it hovers after that

Checking the Knn accuracy again



```
#U19CS022

model = KNeighborsClassifier(n_neighbors = 7)

model.fit(x_train, y_train)
predicted = model.predict(x_test)

print('Confusion Matrix :')
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, predicted))

print()
print()
print("The accuracy of KNN is : ", accuracy_score(y_test, predicted.round())*100, "%")
```

```
Confusion Matrix :
[[23  4]
 [ 3 31]]


The accuracy of KNN is :  88.52459016393442 %
```

Figure 1.32: Checking the Knn accuracy again

The accuracy of KNN is : 88.52459016393442 percentage

1.8 Conclusion

.

1.Most of the models are performing really well.

2.OPTIMIZED KNN is performing the best for the given dataset.(88.5)

With the accuracy of 88.5 percentage our model not can predict the heart attack of patients. Here our data ends were we took a data of heart patients and now with using all the datacycles and machine learning models we are ending this chapter.

Chapter 2

Medical imaging from Data science

Medical imaging, also known as radiology, is the field of medicine in which medical professionals recreate various images of parts of the body for diagnostic or treatment purposes. Medical imaging procedures include non-invasive tests that allow doctors to diagnose injuries and diseases without being intrusive.

Medical imaging is a central part of the improved outcomes of modern medicine. Different types of medical imaging procedures include:

X-rays
Magnetic resonance imaging (MRI)
Ultrasounds
Endoscopy
Tactile imaging
Computerized tomography (CT scan)

Other beneficial medical imaging procedures include nuclear medicine functional imaging techniques, such as positron emission tomography (PET) scans. Other uses of medical imaging include scans to see how well your body is responding to a treatment for a fracture or illness.

2.1 Types of medical data

Medical data (images) are produced by interaction of different forms of radiation with tissue and it can rangefrom a simple chest X-ray to more complex images produced by functional magnetic resonance imaging (fMRI).Different techniques of medical imaging such as radiology, nuclear medicine, or optical imaging, provide imageswith different spatial and temporal resolutions.

2.1.1 Radiography

X-ray is a form of electromagnetic radiation which consists of photons. It was discovered by Wilhelm KonradRongtgen while he was studying cathode tubes. He found out that the tube was emitting light as well as anew mysterious kind of radiation which he called X-rays. Soon he discovered that this radiations can travelthrough different material and also be captured on a photographic plate. Before long, x-rays were being used formedical purposes.9The resolution of the images produced by radiographic systems depends on several parametersincluding the size of the focal spot, thickness of the body part, and the light scattering properties of the fluorescentscreen.10 X-ray photons by nature carry some quantum noise. The noise amplitude is corresponding to thesquare root of the signal amplitude and the signal-to-noise ratio (SNR) behaves as the square root of the signalamplitude. Therefore, dose reduction is not unpunished in image quality. Some conversions during imagingprocess also add noise and reduce the SNR.

2.1.2 X-ray Computed Tomography

Computed tomography (CT) became feasible only after the development of modern computers. It is a tool thatreconstructs images from measured data citect1. Tomographic imaging consists of capturing x-ray images of anobject from multiple orientations and measuring the decrease in intensity along several linear paths. Then, analgorithm reconstructs the distribution of X-ray attenuation in the volume that is scanned.12 CT data consistsof a sequence (thousands) of images which can be visualized using different 2D or 3D image processing tools.Volume rendering and isosurfacing are the two standard modes of 3D visualization of CT data. CT values arethe gray-level numbers in images. Volume rendering involves mapping each CT value to a color and an opacity.Some phases can be rendered transparent, revealing the internal structure. Isosurfacing is defining 3D contoursurfaces distinguishing the boundaries between CT numbers, separating the elevation values on a topo map

2.1.3 Magnetic Resonance Imaging (MRI)

In magnetic resonance imaging (MRI), a powerful magnet to generate images that cannot be captured usingX-rays or CT such as joints, cartilage, ligaments, and tendons.14 The MRI machine is used to create a staticelectromagnetic field to align the proton spins of oxygen atoms in blood. A short radio frequency wave reorientsthe nuclei of the atoms and the atoms absorb this energy. When the interfering wave stops, the protons graduallyreturns to their aligned spin and release the energy that is stored in them. This produces a radio signal thatis measured by the scanner and interpreted into images. Protons in different tissues generate different signalswhich is used to distinguish various types of tissue.15 The MRI data is captured at a very high rate (multipleslices per second). The trade-off for this high speed is a low spatial resolution. MRI data also suffers from avariety of distortions because of the effect of magnetic field inhomogeneities. Furthermore.

2.1.4 Functional MR Imaging (fMRI

Regardless of any external stimuli, live brains show activity unceasingly. The neurons with higher activityconsume more oxygen. Functional MRI is a non-invasive method that locates and measures the fluctuations inblood-oxygen-level dependent fluctuations and provides a map of the functional connectivity in brain.17 Thesignal that is measured is complex valued. Both real and imaginary components are measured with independenterror that is normally distributed. The reconstructed voxel data is also complex valued since Fourier Transform isa linear operation. In most studies, the phase portion is discarded and only magnitude is used since it carries mostof the useful information. It is important to know the behavior of the signal and noise presented in fMRI datato be able to property model the components.18 Neural activity unfolds in time and space. Therefore, spatialand temporal resolution of data can result in some limitations in deriving conclusions. Temporal resolutionaids distinguishing brain events in time and spatial resolution, across spatial locations. The nature of fMRIexperiments prevent having ideal spatial and temporal resolution at the same time. Therefore, it is important tofind the perfect balance between the spatial and temporal resolution requirements for each specific experiment.19MRI and fMRI can use either Arterial Spin Labeling (ASL) or Blood Oxygen Level Dependent (BOLD) signals.The method is selected based on the required sensitivity and other specifications of the experiment

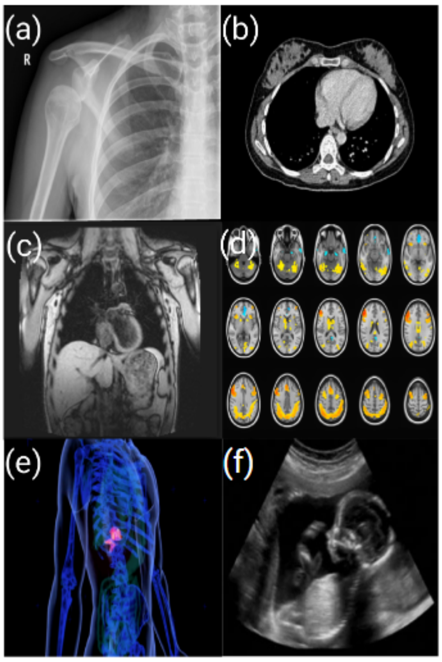


Figure 2.1: scanning images

(f)Figure 2.1. (a) An inferior shoulder dislocation radiology. (b) CT lung cancer screening. (c) Chest MRI. (d) fMRI brainscans of functional connectivity. (e) A sample nuclear medicine image of torso on the human body. (f) Fetal ultrasound.during scans are the other substantial source of noise in MRI data sets. Even slight movements can result in abig change in the captured signal. For this reason, realignment is often one of the primary step in analyzing thisdata

2.2 Nuclear Medicine Imaging

The method of observing the radiation from different parts of the body after a radioactive tracer is injectedor orally given to the patient, is nuclear medicine imaging which is used for observing tumors, infections, andthyroid or bone scintigraphy. It is ensured that the radiation exposure to patients is as low as possible. Thetwo common types of nuclear medicine imaging are Single Photon Emission Computed Tomography (SPECT)and Positron Emission Tomography (PET). PET and SPECT both produce three dimensional images and themain difference is in the radiotracer that is used in the process. Comparing to SPECT, PET is more costlybut it produces better contrast and spatial resolution. PET data can be captured dynamically or statically.Dynamic acquisition lets us observe the long-term behavior of the tracer in the tissue and is a great way to getquantitative measurements of the target area. Static acquisition provides semi-quantitative information and itworks by specifying one time frame over the course of imaging. Static images can also be obtained from dynamicdata by finding the average of radioactivity over a set of time frames.21 In SPECT, the camera moves aroundthe patient and the images are captured from at least 180 degrees. After the scan, reconstruction is done byfiltered-back projection methods. The images are viewed in the transverse, sagittal or coronal planes or as threedimensional models. The useful property of SPECT is that the reconstructed images can be viewed in multipleplanes and it is possible to separate overlapping structures.

2.3 Ultrasound

Ultrasound is a widely available, safe, and non-invasive method for producing real-time images of the structuresinside of the body or the blood flow, by using sound waves. In ultrasound scanning or sonography, high frequencysound waves are transmitted into the body and the transducer collects the reflected signal to create an image.Ultrasound can produce images of thin sections of the body. However, it is possible to create three dimensionalimages from the acquired data

2.4 Analysis

2.4.1 Preprocessing

Medical images, similar to most real-world data, suffer from issues that if not treated, can increase the inaccuracyof the results of analyzing the images. Contrast adjusting, noise reduction, physiological artifacts removal, andhandling the missing data are some of the reasons for performing preprocessing steps, prior to analysis, in orderto validate model assumptions.

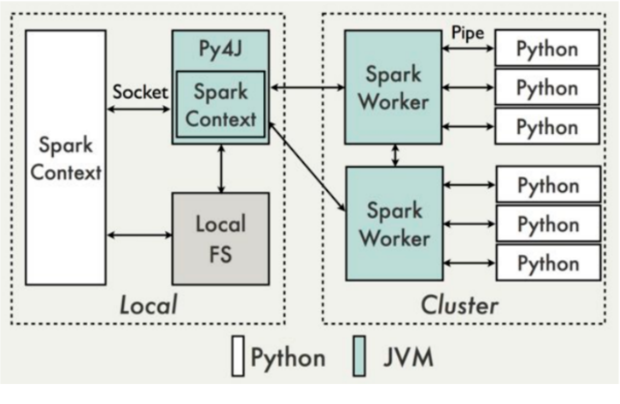


Figure 2.2: physpark dataflow

Figure 2.2. An illustration of PySpark data flow. The Python environments were shaded in white and the Java environmentswere shaded in blue.

2.4.2 segmentation

Segmentation is partitioning the image into sets of regions to extract the areas of interest. Regions can bedefined by a particular shape, border, color, or texture. Classical clustering methods that perform segmentationby finding pixels similar in intensity values, RGB values, texture, and more, include Iterative K-means andIsodate clustering. Histogram methods and different variations of it such as Ohlander's Recursive Histogramtechnique assume that the homogeneous objects in the image can be extracted as clusters on the histogram.31Region growing is the other method for segmentation. In this method, the algorithm starts from one point in theimage (usually the top left corner) and grows the region until the pixels are too different from the current regionand form a set of connected pixels with same population mean and variance.32 The other type of region-basedsegmentation algorithm is Threshold Segmentation which directly ddivides the gray scale information based onthe value of different targets. For images that include several touching objects, the Watershed Segmentationmethods could be applied. The watershed transform seeks catchment basins and watershed ridge lines in animage to distinguish between foreground and background and the region that each pixel belongs to

2.4.3 Region of Interest (ROI)

ROI analysis involves extracting the signal from specified regions by selecting clusters of pixels or voxels in theimage and it can be used for analysis withing one subject or across multiple ones. Using ROI techniques reducesthe type I errors that can occur in analysis, by limiting the number of statistical tests to a few ROIs.34 Inanalyzing medical images, loss of data results in loss of vital information. ROI is mainly used for medical images.Each image is divided into two parts, foreground; the areas that carry diagnostically important information, andbackground; the rest of the image. To preserve the quality of the diagnostic part, lossless compression techniquesare favorable. Additionally, the diagnostic part has higher priority in transmissions.35 Exploratory ROIs arespheres of the same diameter, placed at the local maxima in the statistical map. The locations of the ROIscan be selected based on anatomical templates such as Talairach atlas for brains, or functionally based on thedata from images produced by techniques such as fMRI, or based on previous studies.34 For different types ofmedical images, there are several tools that help with placing and analyzing ROIs such as SPM by WellcomeTrust Centre for Neuroimaging for brain images or Matlab for different types of medical images. After extractingall the ROI coordinates, a value should be calculated for each point of interest. The simplest way of calculatingthis value is finding the mean for each point. However, the mean can be easily affected by outliers in which case.

2.4.4 Fourier Transform

The Fourier Transform is suitable for image processing including filtering, compression, and reconstruction, todecompose the image into sine and cosine components which represent the image. The Discrete Fourier Transform(DFT) provides a sample of all frequencies in the image that is large enough to fully represent the geometric-characteristics of a spatial domain image. DFT can provide a good representation of signal changes and behaviorfor discrete time signals.42 The characteristics that change with time cannot be represented using DFT since itcan only be used for slices (windows) of the signals that have a fixed time duration

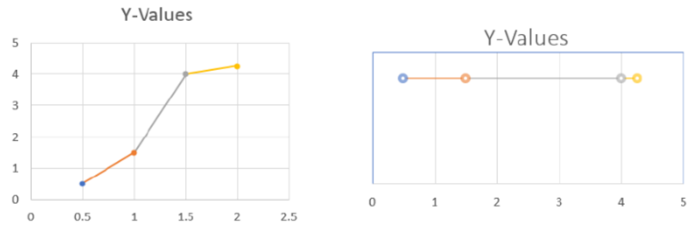


Figure 2.3: PCA Transformation

Figure 2.3. PCA Transformation of a 2D (left) to a 1D line graph (right) - In this dataset, the value of X is barely informativesince the distances are regular. By removing the X values and projecting the Y values onto a 1D chart, we can obervethe variances more visibly

Bibliography

[1] Tahmassebi, A., “ideeple: Deep learning in a flash,” in [Disruptive Technologies in Information Sciences],10652, 106520S, International Society for Optics and Photonics (2018).

[2] Webb, A. and Kagadis, G. C., “Introduction to biomedical imaging,” Medical Physics 30(8), 2267–2267(2003)

[3] kaggle website

[4] Big data analytics in medical imaging using deep learning Amirhessam Tahmassebi Florida State University Amir H Gandomi University of Technology Sydney

[5] Geeks for Geeks , Tutorial point and other learning websites.