EXPLORATORY DATA ANALYSIS ON HEART FAILURE

ENGINEERING CLINIC PROJECT REPORT

Submitted by

LASHYA P(18BCS045)
BHRAMI.R.R(18BCS056)
SREE NANDHINI(18BCS051)
KAAVIYA HARANI.M(18BCS202)
AHALYA B(18BCS204)
SAMRITHA.N(18BCS205)

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

COIMBATORE-641 049

(An Autonomous Institution Affiliated to Anna University, Chennai)

December 2020

TABLE OF CONTENTS

ABSTRACT

Healthcare industries generate enormous amount of data, so called big data that accommodates hidden knowledge or pattern for decision making. Exploratory Data Analysis (EDA) detects mistakes, finds appropriate data, checks assumptions and determines the correlation among the explanatory variables. In the context, EDA is considered as analysing data that excludes inferences and statistical modelling. Analytics is an essential technique for any profession as it forecast the future and hidden pattern. Data analytics is considered as a cost effective technology in the recent past and it plays an essential role in healthcare which includes new research findings, emergency situations and outbreaks of disease. The use of analytics in healthcare improves care by facilitating preventive care and EDA is a vital step while analysing data. In this paper, the risk factors that causes heart disease is considered and predicted using K-means algorithm and the analysis is carried out using a publicly available data for heart disease. The dataset holds 209 records with 8 attributes such as age, chest pain type, blood pressure, blood glucose level, ECG in rest, heart rate and four types of chest pain. To predict the heart disease, K-means clustering algorithm is used along with data analytics and visualization tool. The paper discusses the pre-processing methods, classifier performances and evaluation metrics. In the result section, the visualized data shows that the prediction is accurate

1.INTRODUCTION

Predicting and diagnosing heart disease is the biggest challenge in the medical industry and it is based on factors like physical examination, symptoms and signs of the patient . Factors which influence heart diseases are cholesterol level of the body, smoking habit, and obesity, family history of diseases, blood pressure and working environment. Machine learning algorithms play a vital and accurate role in predicting heart disease. The advancement of technologies allows machine language to pair with big data tools to handle unstructured and exponentially growing data . In the paper, K means clustering method is proposed in big data environment and the visualization is made with the tableau dashboard.

1.1 CONCEPTUAL STUDY OF THE PROJECT

Exploratory data analysis is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task. Exploratory data analysis was promoted by John Tukey to encourage statisticians to explore the data, and possibly formulate hypotheses that could lead to new data collection and experiments. EDA is different from initial data analysis(IDA), which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing, and handling missing values and making transformations of variables as needed. EDA encompasses IDA. The primary **goal of EDA** is to maximize the analyst's insight into a data set and into the underlying structure of a data set, while providing all of the specific items that an analyst would want to extract from a data set, such as: a good-fitting, parsimonious model. a list of outliers.

1.2 OBJECTIVE OF THE PROJECT

To predict mortality by heart failure by combining certain attributes.

1.3 SCOPE OF THE PROJECT

Besides adhering to existing guidelines, newer approaches to managing these patients, both in terms of monitoring and

developing novel therapeutic approaches, are needed.

Significant opportunities exist to improve the outcomes for patients with HF, especially those who have been hospitalized.

These efforts are even more important now that readmission rates for HF have quality and reimbursement implications

2. PROBLEM DEFINITION

Heart disease can be managed effectively with a combination of lifestyle changes, medicine and, in some cases, surgery. With the right treatment, the symptoms of heart disease can be reduced and the functioning of the heart improved. The predicted results can be used to prevent and thus reduce cost for surgical treatment and other expensive.
The overall objective of my work will be to predict accurately with few tests and attributes the presence of heart disease. Attributes considered form the primary basis for tests and give accurate results more or less. Many more input attributes can be taken but our goal is to predict with few attributes and faster efficiency the risk of having heart disease.

3. LITERATURE REVIEW

3.1 LITERATURE REVIEW OF JOURNALS

The American Journal of Accountable Care. 2017;5(3):21-25

Heart failure (HF) is a complex, clinical syndrome of signs and symptoms that are caused by defects in cardiac structure, function, or both, resulting in impairment of peripheral circulation and organ oxygenation. More than 6 million people in the United States older than 20 years have HF; approximately 800,000 new cases are diagnosed each year. HF prevalence is increasing, with estimates indicating that more than 8 million people in the United States may have HF by 2030.

HF is a major public health problem associated with high morbidity and mortality among individuals 65 years and older. It is the most common principal discharge diagnosis among Medicare beneficiaries and the third highest for hospital reimbursements. In the single year of 2007, there were 1.4 million hospitalizations, more than 11 million office visits, and $17 billion in total spending attributable to HF alone. The progressive and complicated nature of this disease, coupled with multiple comorbidities, often results in negative outcomes, the most costly being hospital readmissions. Approximately 25% of patients admitted due to HF are readmitted within 30 days, and 34% are readmitted within 90 days of discharge. Recent CMS data showed that unplanned HF 30-day readmission rates showed a slight downward trend from 2009 to 2012. However, postdischarge emergency department (ED) and observation unit visits increased.

HF hospital readmissions, including ED visits, are considered preventable. The majority of HF patients are often discharged early in the recovery period with inadequate self-care instructions, poor management of the underlying problems, and poor multidisciplinary coordination. To address the adverse effects on quality of care and healthcare costs of HF readmissions (as well as other readmissions), CMS implemented the Hospital Readmission Reduction Program, which reduced Medicare base reimbursements to underperforming hospitals by 1% in 2013, 2% in 2014, and 3% in 2015.9 This affects the inpatient services provided for all diagnosis-related group readmissions within 30 days of an HF admission. The aging population of the United States, combined with increasing chronic comorbidities, justifies the need for strategies and innovations to address the major gaps in transitions of care across healthcare settings.

A transition in care is defined as the time when a patient transfers from a hospital to home or community. Transition points are considered vulnerable times that contribute to high healthcare costs as well as gaps in quality and safety of care; they are associated with hospital readmissions. The American Geriatrics Society defines a transition of care plan as a group of multidisciplinary actions designed to ensure coordination and continuity of care when patients are transferred between different levels of care or locations. Ideally, a thoughtful transition of care should begin during admission and continue through the patient's return home. It should contain the element of communication between providers to ensure continuity of care.

Innovative interventions stand out in improving the continuity of care for patients with HF across episodes of care. Called HF transitional care interventions or programs, they are designed to prevent hospital readmissions and include a substantial number of time-limited interventions that help establish continuity of care, avoid preventable poor outcomes, and ensure the safe and timely transfer of patients with complex chronic conditions from one level of care to another or from one type of setting to another.

The transitional care model (TCM) was designed by Mary Naylor, PhD, MSN, and a multidisciplinary team of colleagues at the University of Pennsylvania in Philadelphia. It addresses the detrimental effects associated with the usual breakdowns in care when older adults with complex needs transition from an acute hospital setting to their home or community. This model has been tested and refined for the past 20 years. TCM has essential core elements that complement one another, establishing a strong foundation critical to the successful implementation of TCM programs. Two of the core elements of TCM are the utilization of a transitional care nurse with advanced education, skills, and knowledge in coordinating the care of high-risk older adults in different healthcare settings by using a multidisciplinary approach, as well as strong collaboration between patient, family/caregivers, and members of the healthcare team.

An array of HF transitional care interventions is being utilized to improve the quality of care and outcomes, and, eventually, reduce healthcare costs. Transitional care interventions that have proven success in reducing hospital readmissions are important components of healthcare reform. There is a critical need to replicate successful programs that build strong connections between current evidence-based practices and approaches to care.

The objective of this literature review was to evaluate the effectiveness of different transitional care interventions used in inpatient and outpatient settings that attempt to reduce hospital readmissions.

Transitional Care Interventions

In a systematic review conducted by Feltner et al, 47 RCTs were compared and assessed on the effectiveness and efficacy of HF transitional care interventions in reducing readmission and mortality rates among hospitalized elderly patients with HF. Each of the studies implemented at least 1 HF transitional care intervention or a combination of different transitional care interventions and compared it with usual care. The transitional care interventions used included patient and caregiver HF education provided before and after discharge, scheduled outpatient clinic visits within a week after discharge with the primary care provider (PCP) or HF multidisciplinary clinic, in-person home visits, systematic telephone support (STS) and follow-up, use of transition coach or case manager, and telemonitoring. This study showed that home visits and STS interventions both reduced the risk for HF-specific readmissions. A multidisciplinary HF outpatient clinic, in-person home visiting, and systematic telephone follow-up decreased all-cause readmissions within 3 to 6 months after an initial hospitalization. Telemonitoring did not reduce the risk of HF-specific readmissions.

A limitation of this study was that the RCTs did not report effects of interventions on 30-day readmission rates. Feltner et al recommended that in order to evaluate the true effectiveness of interventions, future studies should directly compare one intervention with another and evaluate whether interventions that reduce readmission rates over 3 to 6 months will also reduce 30-day readmission rates. The study design and the use of appropriate data analysis, such as the use of calculated risk ratios and number needed to treat for readmission and mortality rates and outcomes, contributed to the strength of this study.

Delivery Mode of Transitional Care Interventions

When it comes to mode of delivery and communication used in implementing transitional care interventions, Sochalski et al3 evaluated data from 10 RCTs of HF care management programs conducted from 1990 through 2004 in the United States, Australia, the Netherlands, and the United Kingdom. Regression analyses using linear mixed models were used to capture the fixed effects of delivery method elements. Random effects of the 10 parent trials were used to evaluate the effects of delivery method of communication on hospital readmissions and readmission days. Results showed that HF programs with in-person (face-to-face) communication achieved a significant reduction in readmissions and readmission days compared with routine care of patients and with programs using telephonic communication. Programs using only single HF experts were less effective in reducing hospital readmissions compared with those using multidisciplinary teams, regardless of the mode of communication used. The most significant finding of this study was that patients in the transitional care program that offered the combination of a multidisciplinary team approach and in-person communication had significantly fewer hospital readmissions (3%) and readmission days (6%) per month than routine-care patients. The design, appropriate use of data analysis, and large sample size added to the strength of evidence.

Advanced Practice Nurse—Led HF Programs

Naylor et al conducted a systematic review of 21 RCTs. Most of the interventions shared 1 similar component, which was designating an advanced practice nurse as the leader and manager of the entire HF program. This specific element of the HF program was combined with other groups of interventions, such as comprehensive discharge planning and follow-up with or without home visits, coaching, patient education, peer support, telehealth facilitation, mobile crisis, and postdischarge follow-up with the PCP. These interventions demonstrated positive effects in reducing hospital readmissions through at least 30 days after discharge.

Inpatient HF Program

The reengineered hospital discharge (RED) program, a program at the core of an RCT conducted by Jack et al,17 is currently being utilized by most hospitals in their HF programs. This program has a proven track record in successfully decreasing hospital readmissions overall. The 3 essential elements of the initial project RED included a nurse discharge advocate (DA), an after-hospital care plan (AHCP), and a follow-up phone call by the pharmacist. DAs carried out all of the different facets of the hospital interventions, including the RED interventions. Moreover, the DAs designed the AHCP, coordinated and disseminated the discharge plan with the hospital team, and provided HF education to the patient and family/caregiver before and during discharge from the hospital. The clinical pharmacist used the AHCP and hospital visit summary and telephoned the patients 2 to 4 days after the initial discharge to reiterate the patient-specific home discharge plan. Any medication-related issues were communicated to the DA or patient PCP.

Findings from this study showed that the RED intervention decreased hospital utilization within 30 days of discharge by about 30% (utilization was calculated based on combined ED visits and hospital readmissions). Furthermore, patients reported seeing their PCP for follow-up within 30 days and reported a higher level of preparedness for discharge. No other significant studies have investigated these 3 interventions bundled together. Therefore, additional future research should be conducted upon programs that implement these 3 interventions bundled together. The only significant limitation of the study is that the patient population was not exclusively those with HF.

Linden and Butterworth conducted an RCT in 2 independent, nonprofit hospitals. A comprehensive inpatient transitional care program to reduce readmission for patients with congestive heart failure and chronic obstructive pulmonary disease was examined. A total of 257 patients with congestive heart failure participated in this study. The HF transitional care program included 3 sets of comprehensive components commonly found in the transitional care model. Predischarge components included patient education, discharge planning, medication reconciliation, and follow-up appointment schedules. Postdischarge components were a timely follow-up telephone call and the availability of a patient hotline. Bridging components included a transition/health coach providing patient-centered discharge instructions. In addition to these components, 2 postdischarge interventions were added. These included motivational interviewing-based health coaching and symptom monitoring using interactive voice response. Due to limited staff resources and funding limitations, home visits and provider continuity of care, which are elements often included in TCMs, were not included.

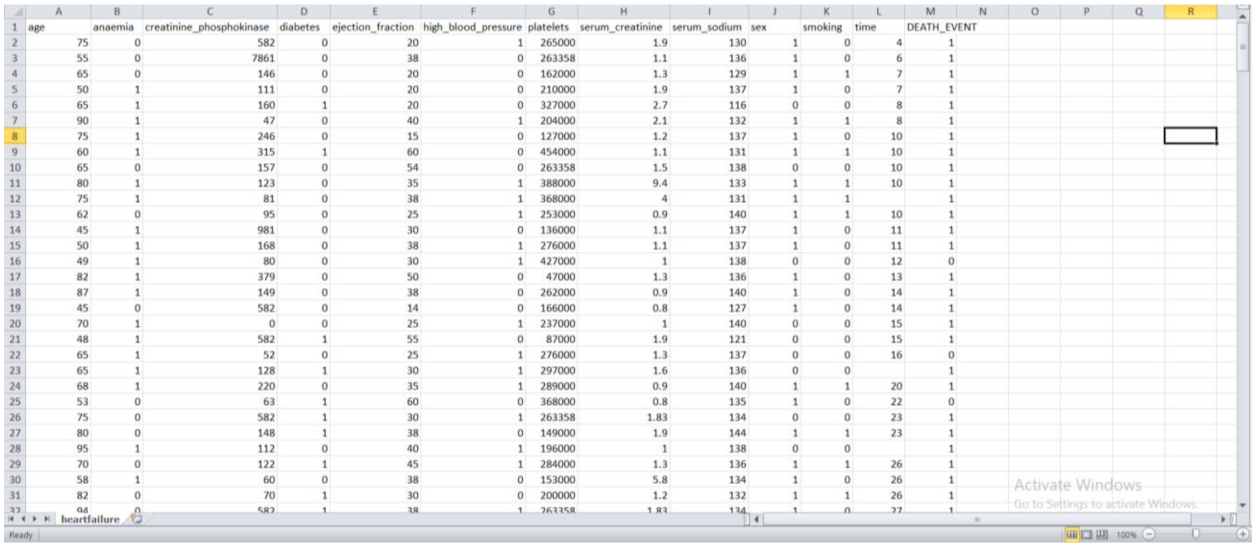
Results of this study showed that there were no statistical differences in either 30-day or 90-day readmission incidence rates. The 30-day readmission rates were 0.23 per person for the intervention group and 0.19 per person for the usual care group (difference = 0.04; 95% CI, —0.05 to 0.13; P = .36). The 90-day readmission rates were 0.51 and 0.48 per person for the intervention and usual care groups, respectively (difference = 0.04; 95% CI, 0.12-0.19; P = .66). Despite the comprehensiveness of the interventions used and high number of participants, this study failed to reduce 30- to 60-day HF readmission rates among patients hospitalized for HF. This could be due to the exclusion of the aforementioned components, home visits and timely follow-up care, in this transitional care program.

Outpatient HF Program

Wakefield et al analyzed different HF interventions that were used in multicomponent outpatient HF programs, examining frequency of the interventions' use, content of the interventions, and effects on patient outcomes. This systematic review included 35 RCTs. The outpatient interventions included face-to-face visits of patients with providers and nurses. Nursing staff services included home visits, remote monitoring of vital signs, remote videophone and messaging, and telephone calls. The majority of these interventions had teaching components, which covered such topics as early recognition and management of HF signs and symptoms, home medication review, and self-monitoring. Findings of this study showed a significant reduction in readmission rates ($P <.001$), better quality of life ($P <.007$), and lower healthcare cost ($P = .008$) among the treatment group compared with the control group. Aside from the design of the study as 1 of its strength, this study provided the most detailed analysis to date of the individual components of multicomponent outpatient HF programs.

4. EXPLORATORY DATA ANALYSIS

HEART FAILURE CLINICAL RECORD DATASET:



age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time	DEATH_EVENT
75	0	582	0	20	1	265000	1.9	130	1	0	4	1
55	0	7861	0	38	0	263358	1.1	136	1	0	6	1
65	0	146	0	20	0	162000	1.3	129	1	1	7	1
50	1	111	0	20	0	210000	1.9	137	1	0	7	1
65	1	160	1	20	0	327000	2.7	116	0	0	8	1
90	1	47	0	40	1	204000	2.1	132	1	1	8	1
75	1	246	0	15	0	127000	1.2	137	1	0	10	1
60	1	315	1	60	0	454000	1.1	131	1	1	10	1
65	0	157	0	54	0	263358	1.5	138	0	0	10	1
80	1	123	0	35	1	388000	9.4	133	1	1	10	1
75	1	81	0	38	1	368000	4	131	1	1		1
62	0	95	0	25	1	253000	0.9	140	1	1	10	1
45	1	981	0	30	0	136000	1.1	137	1	0	11	1
50	1	168	0	38	1	276000	1.1	137	1	0	11	1
49	1	80	0	30	1	427000	1	138	0	0	12	0
82	1	379	0	50	0	47000	1.3	136	1	0	13	1
87	1	149	0	38	0	262000	0.9	140	1	0	14	1
45	0	582	0	14	0	166000	0.8	127	1	0	14	1
70	1	0	0	25	1	237000	1	140	0	0	15	1
48	1	582	1	55	0	87000	1.9	121	0	0	15	1
65	1	52	0	25	1	276000	1.3	137	0	0	16	0
65	1	128	1	30	1	297000	1.6	136	0	0		1
68	1	220	0	35	1	289000	0.9	140	1	1	20	1
53	0	63	1	60	0	368000	0.8	135	1	0	22	0
75	0	582	1	30	1	263358	1.83	134	0	0	23	1
80	0	148	1	38	0	149000	1.9	144	1	1	23	1
95	1	112	0	40	1	196000	1	138	0	0		1
70	0	122	1	45	1	284000	1.3	136	1	1	26	1
58	1	60	0	38	0	153000	5.8	134	1	0	26	1
82	0	70	1	30	0	200000	1.2	132	1	1	26	1
94	0	582	1	38	1	263358	1.83	134	1	0	27	1

heartfailure

Ready

4.1 ABOUT THE DATASET

- Cardiovascular diseases (CVDs) are the **number 1 cause of death globally**, taking an estimated **17.9 million lives each year**, which accounts for **31% of all deaths worlwide**.
- Heart failure is a common event caused by CVDs and this dataset contains 12 features that can be used to predict mortality by heart failure.
- Most cardiovascular diseases can be prevented by addressing behavioural risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol using population-wide strategies.
- People with cardiovascular disease or who are at high cardiovascular risk (due to the presence of one or more risk factors such as hypertension, diabetes, hyperlipidaemia or already established disease) need **early detection** and management wherein we are analysing this dataset.

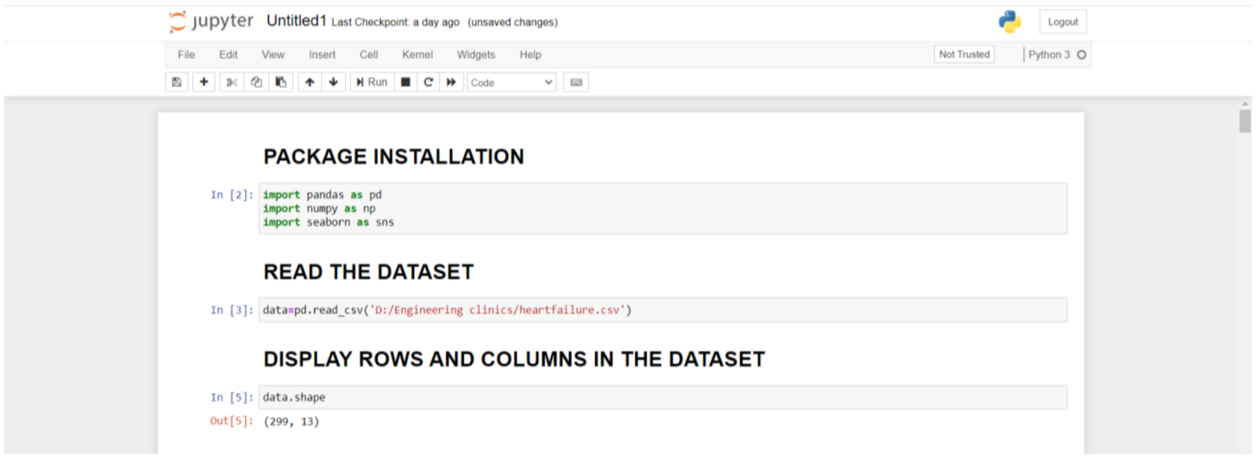
4.2 UNDERSTANDING THE VARIABLES

- Age-age of person
- Anaemia-Decrease of red blood cells or hemoglobin (boolean)
- Creatinine_phosphokinase-Level of the CPK enzyme in the blood (mcg/L)
- Diabetes-If the patient has diabetes (boolean)

- Ejection_fraction-Percentage of blood leaving the heart at each contraction (percentage)
- High_blood_presure-If the patient has hypertension (boolean)
- Platelets-Platelets in the blood (kiloplatelets/mL)
- Serum_creatinine-Level of serum creatinine in the blood (mg/dL)
- Serum_sodium-Level of serum sodium in the blood (mEq/L)
- Sex-Woman or man (binary)
- Smoking-If the patient smokes or not (boolean)
- Time-follow-up period(days)
- Death_event-if patient deceased during follow-up period(boolean)

4.3 ANALYSIS OF VARIABLES

- CATEGORICAL
  - Diabetes
  - Sex
  - Smoking
  - high_blood_pressure
  - death_event
  - Anaemia
- DISCRETE
  - Age
  - time
- CONTINUOUS
  - Serum_creatinine
  - Serum_sodium
  - Platelets
  - Ejection_fraction
  - Creatinine_phosphokinase



PACKAGE INSTALLATION

```
In [2]: import pandas as pd
        import numpy as np
        import seaborn as sns
```

READ THE DATASET

```
In [3]: data=pd.read_csv('D:/Engineering clinics/heartfailure.csv')
```

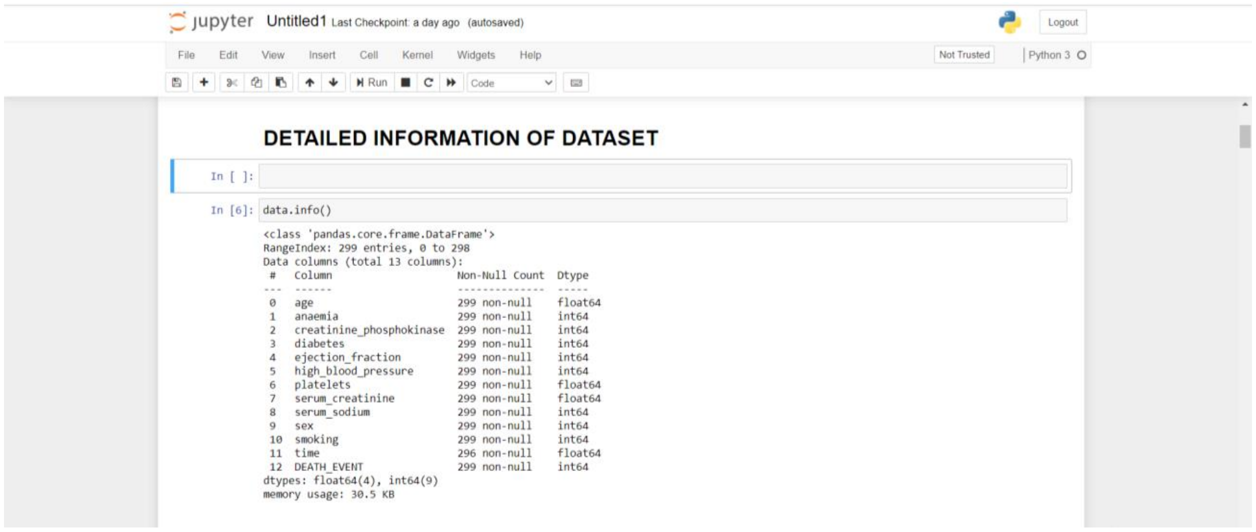
DISPLAY ROWS AND COLUMNS IN THE DATASET

```
In [5]: data.shape
Out[5]: (299, 13)
```

In the above picture, 1st we install 3 packages such that pandas, numpy and seaborn

Then we read the dataset using that second command.

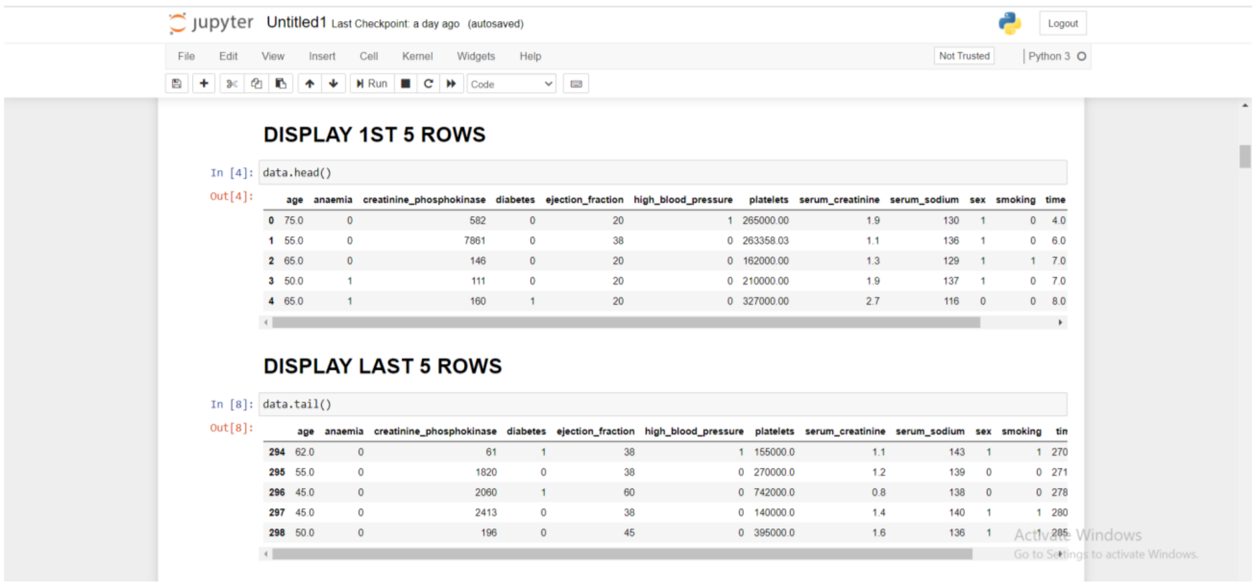
By using shape command, we came to know about the number of rows and column in the dataset.



DETAILED INFORMATION OF DATASET

```
In [ ]:

In [6]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 299 entries, 0 to 298
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   age                       299 non-null    float64
 1   anaemia                   299 non-null    int64  
 2   creatinine_phosphokinase  299 non-null    int64  
 3   diabetes                  299 non-null    int64  
 4   ejection_fraction         299 non-null    int64  
 5   high_blood_pressure       299 non-null    int64  
 6   platelets                 299 non-null    float64
 7   serum_creatinine          299 non-null    float64
 8   serum_sodium              299 non-null    int64  
 9   sex                       299 non-null    int64  
 10  smoking                   299 non-null    int64  
 11  time                      296 non-null    float64
 12  DEATH_EVENT               299 non-null    int64  
dtypes: float64(4), int64(9)
memory usage: 30.5 KB
```

By using info() command, we get the data type for all the columns



jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

DISPLAY 1ST 5 ROWS

```
In [4]: data.head()
Out[4]:
```

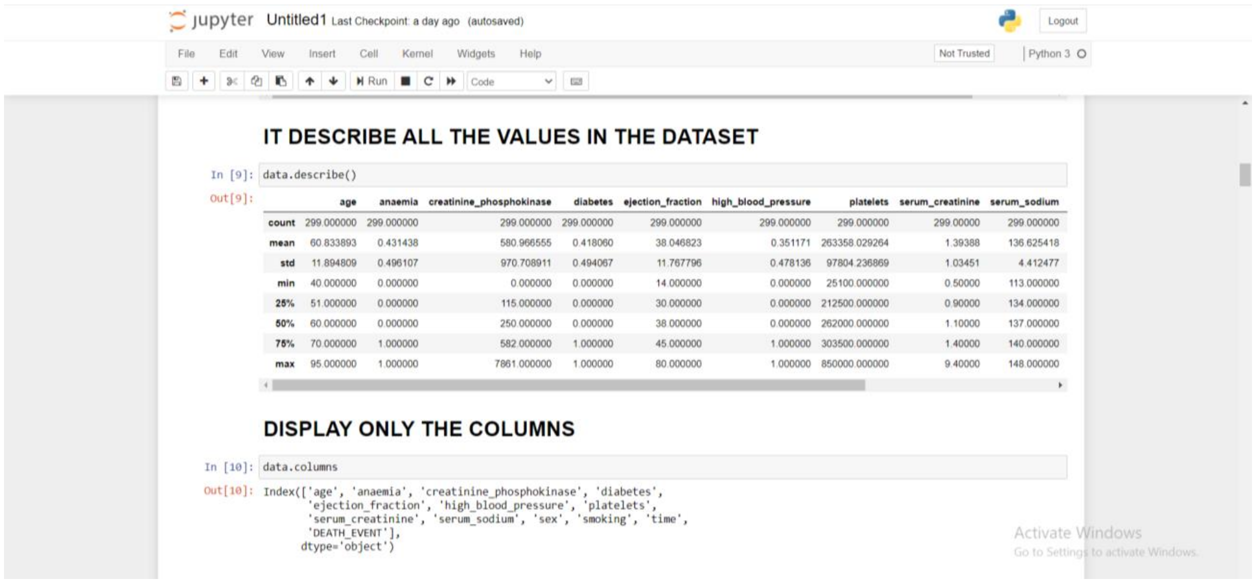
	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	time
0	75.0	0	582	0	20	1	265000.00	1.9	130	1	0	4.0
1	55.0	0	7861	0	38	0	263358.03	1.1	136	1	0	6.0
2	65.0	0	146	0	20	0	162000.00	1.3	129	1	1	7.0
3	50.0	1	111	0	20	0	210000.00	1.9	137	1	0	7.0
4	65.0	1	160	1	20	0	327000.00	2.7	116	0	0	8.0

DISPLAY LAST 5 ROWS

```
In [8]: data.tail()
Out[8]:
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	tin
294	62.0	0	61	1	38	1	155000.0	1.1	143	1	1	270
295	55.0	0	1820	0	38	0	270000.0	1.2	139	0	0	271
296	45.0	0	2060	1	60	0	742000.0	0.8	138	0	0	278
297	45.0	0	2413	0	38	0	140000.0	1.4	140	1	1	280
298	50.0	0	196	0	45	0	395000.0	1.6	136	1	1	285

Head() command displays first 5 rows in the dataset
Tail() command displays last 5 rows in the dataset



jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

IT DESCRIBE ALL THE VALUES IN THE DATASET

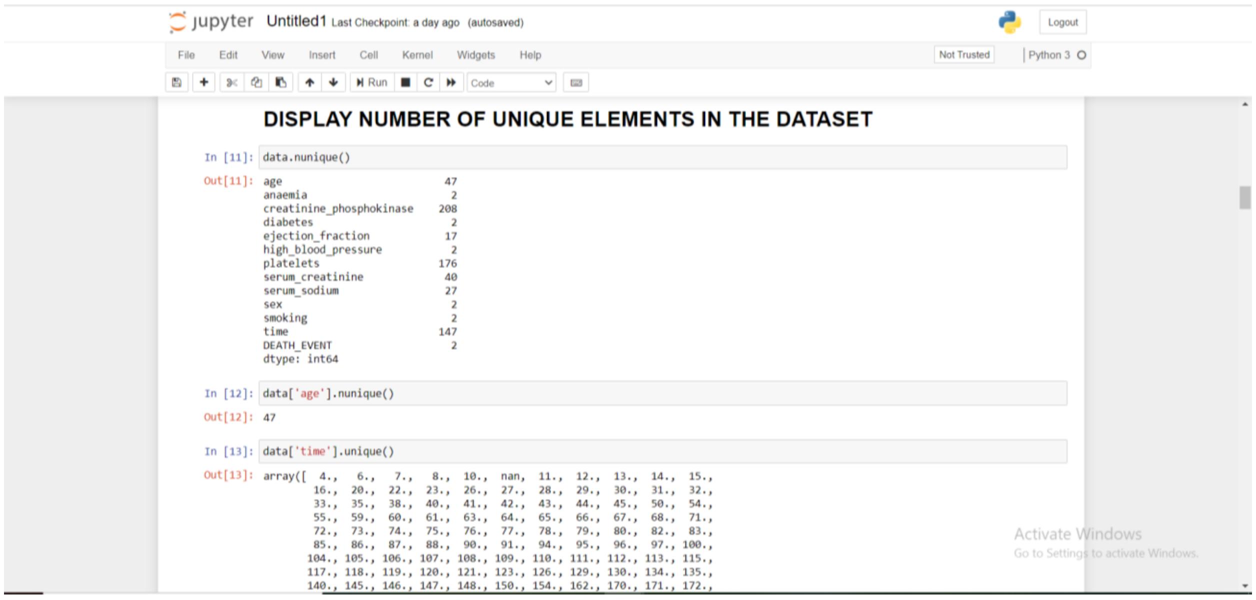
```
In [9]: data.describe()
Out[9]:
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium
count	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.00000	299.000000
mean	60.833893	0.431438	580.966555	0.418060	38.046823	0.351171	263358.029264	1.39388	136.625418
std	11.894809	0.496107	970.708911	0.494067	11.767796	0.478136	97804.236869	1.03451	4.412477
min	40.000000	0.000000	0.000000	0.000000	14.000000	0.000000	25100.000000	0.50000	113.000000
25%	51.000000	0.000000	115.000000	0.000000	30.000000	0.000000	212500.000000	0.90000	134.000000
50%	60.000000	0.000000	250.000000	0.000000	38.000000	0.000000	262000.000000	1.10000	137.000000
75%	70.000000	1.000000	582.000000	1.000000	45.000000	1.000000	303500.000000	1.40000	140.000000
max	95.000000	1.000000	7861.000000	1.000000	80.000000	1.000000	850000.000000	9.40000	148.000000

DISPLAY ONLY THE COLUMNS

```
In [10]: data.columns
Out[10]: Index(['age', 'anaemia', 'creatinine_phosphokinase', 'diabetes',
                'ejection_fraction', 'high_blood_pressure', 'platelets',
                'serum_creatinine', 'serum_sodium', 'sex', 'smoking', 'time',
                'DEATH_EVENT'],
               dtype='object')
```

Describe() command displays the dataset by splitting the categories like mean, count, standard deviation, minimum, maximum, 25%, 50%, 75%, we know that **column age has minimum value of 40, maximum value of 95 and so on**.
data.columns, this command display all the columns in the dataset, we known that there are **13 columns** in our dataset.

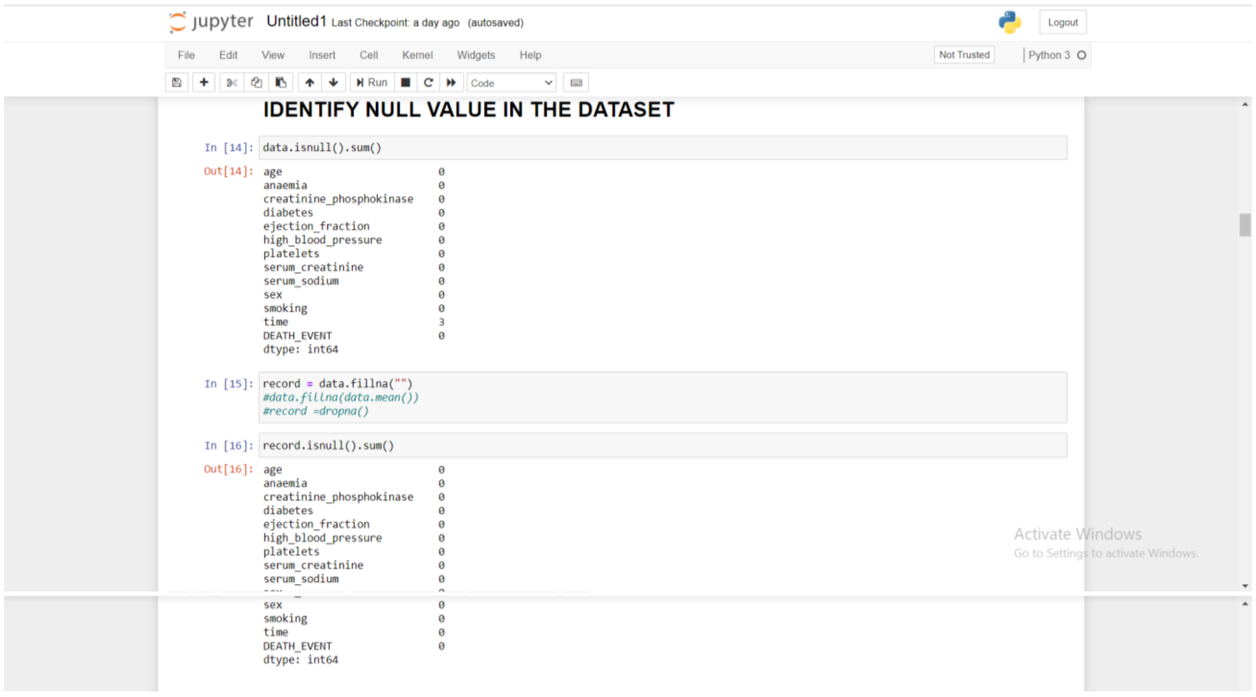


DISPLAY NUMBER OF UNIQUE ELEMENTS IN THE DATASET

```
In [11]: data.nunique()

Out[11]: age                          47
         anaemia                       2
         creatinine_phosphokinase    208
         diabetes                      2
         ejection_fraction            17
         high_blood_pressure           2
         platelets                   176
         serum_creatinine             40
         serum_sodium                 27
         sex                           2
         smoking                       2
         time                        147
         DEATH_EVENT                   2
         dtype: int64

In [12]: data['age'].nunique()

Out[12]: 47

In [13]: data['time'].unique()

Out[13]: array([  4.,   6.,   7.,   8.,  10.,  nan,  11.,  12.,  13.,  14.,  15.,
                 16.,  20.,  22.,  23.,  26.,  27.,  28.,  29.,  30.,  31.,  32.,
                 33.,  35.,  38.,  40.,  41.,  42.,  43.,  44.,  45.,  50.,  54.,
                 55.,  59.,  60.,  61.,  63.,  64.,  65.,  66.,  67.,  68.,  71.,
                 72.,  73.,  74.,  75.,  76.,  77.,  78.,  79.,  80.,  82.,  83.,
                 85.,  86.,  87.,  88.,  90.,  91.,  94.,  95.,  96.,  97., 100.,
                104., 105., 106., 107., 108., 109., 110., 111., 112., 113., 115.,
                117., 118., 119., 120., 121., 123., 126., 129., 130., 134., 135.,
                140., 145., 146., 147., 148., 150., 154., 162., 170., 171., 172.,
```

nunique() displays the number of unique values in the each column, we came to know that **column age has 43 unique value, column anaemia as 2 unique** values and so on.

4.3 MISSING VALUE TRATMENT

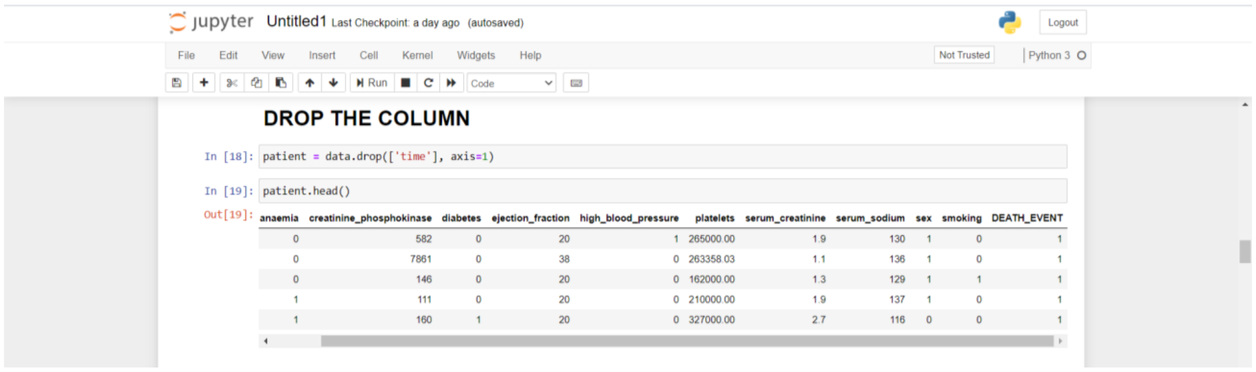


IDENTIFY NULL VALUE IN THE DATASET

```
In [14]: data.isnull().sum()

Out[14]: age                         0
         anaemia                     0
         creatinine_phosphokinase    0
         diabetes                    0
         ejection_fraction           0
         high_blood_pressure         0
         platelets                   0
         serum_creatinine            0
         serum_sodium                0
         sex                         0
         smoking                     0
         time                        3
         DEATH_EVENT                 0
         dtype: int64

In [15]: record = data.fillna("")
         #data.fillna(data.mean())
         #record =dropna()

In [16]: record.isnull().sum()

Out[16]: age                         0
         anaemia                     0
         creatinine_phosphokinase    0
         diabetes                    0
         ejection_fraction           0
         high_blood_pressure         0
         platelets                   0
         serum_creatinine            0
         serum_sodium                0
         sex                         0
         smoking                     0
         time                        0
         DEATH_EVENT                 0
         dtype: int64
```

isnull() command is used to identify the null values in the dataset, we came to know that **column time has 3 null** values.

fillna() command is used to fill the null datasets. Here we **fill our dataset** using **the empty space.**
In last output we see that the null values can be filled.

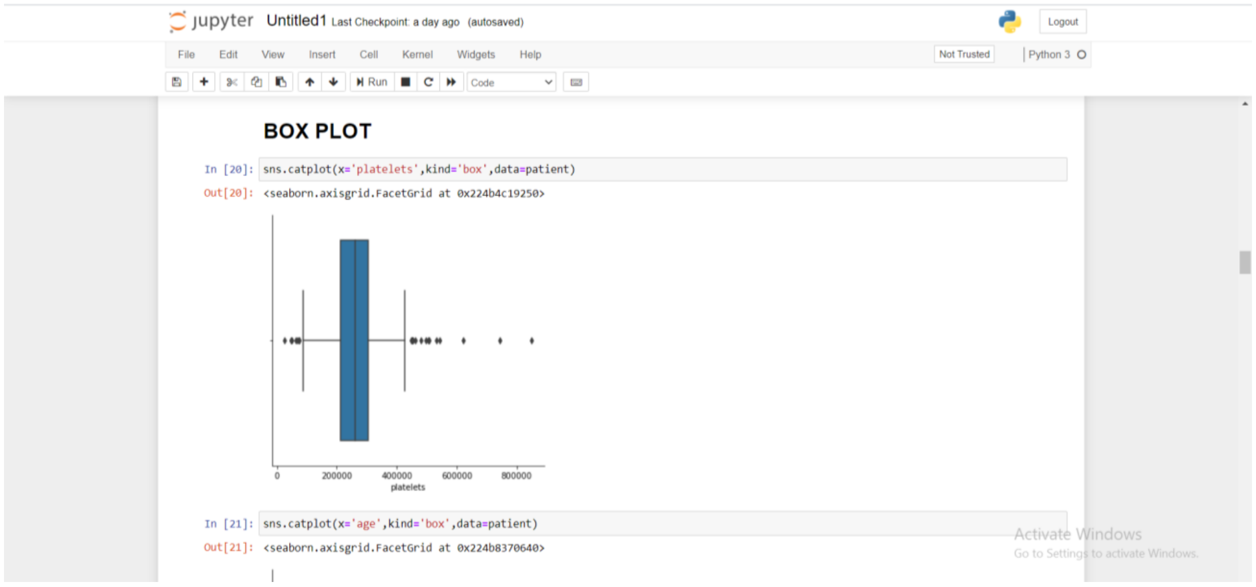

jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

DROP THE COLUMN

```
In [18]: patient = data.drop(['time'], axis=1)

In [19]: patient.head()

Out[19]:
```

anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	DEATH_EVENT
0	582	0	20	1	265000.00	1.9	130	1	0	1
0	7861	0	38	0	263358.03	1.1	136	1	0	1
0	146	0	20	0	162000.00	1.3	129	1	1	1
1	111	0	20	0	210000.00	1.9	137	1	0	1
1	160	1	20	0	327000.00	2.7	116	0	0	1

By using drop command we can drop unnecessary column in our dataset, in this we see that **time column** is **dropped**.

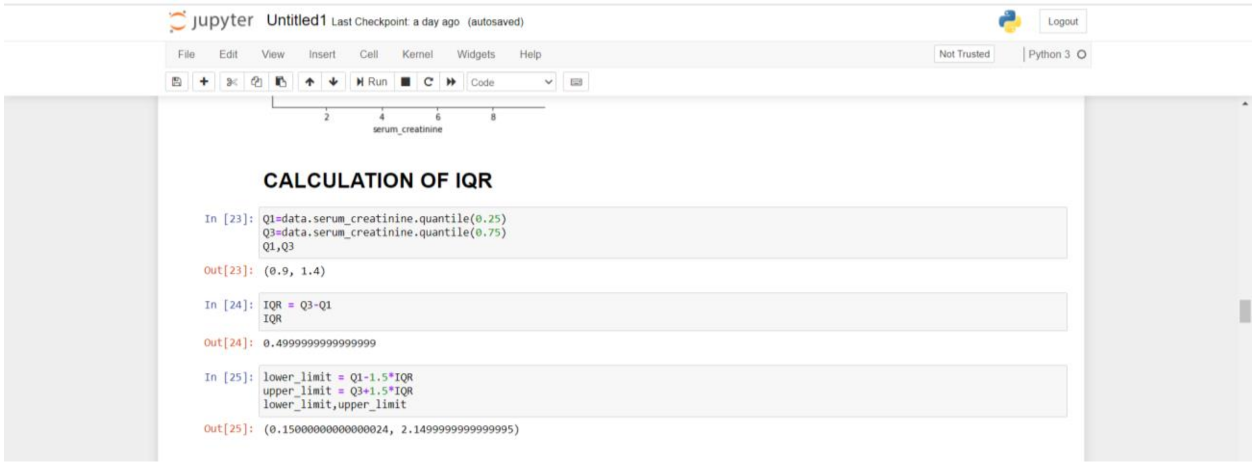

jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
0
200000
400000
600000
800000
platelets
Activate Windows
Go to Settings to activate Windows.

BOX PLOT

```
In [20]: sns.catplot(x='platelets',kind='box',data=patient)
Out[20]: <seaborn.axisgrid.FacetGrid at 0x224b4c19250>

In [21]: sns.catplot(x='age',kind='box',data=patient)
Out[21]: <seaborn.axisgrid.FacetGrid at 0x224b8370640>
```

By using box-plot, we can find the outliers (ie. Outliers means out of range in IQR)
In that blue box, the **left side line** indicates the range of **25% of platelets**, **right side line** indicates the range of **75% of platelets**, **middle line** indicate the **mean value.**
The **left most line** indicates the range of the **lower limit.**
The **right most line** indicates the range of the **upper limit.**

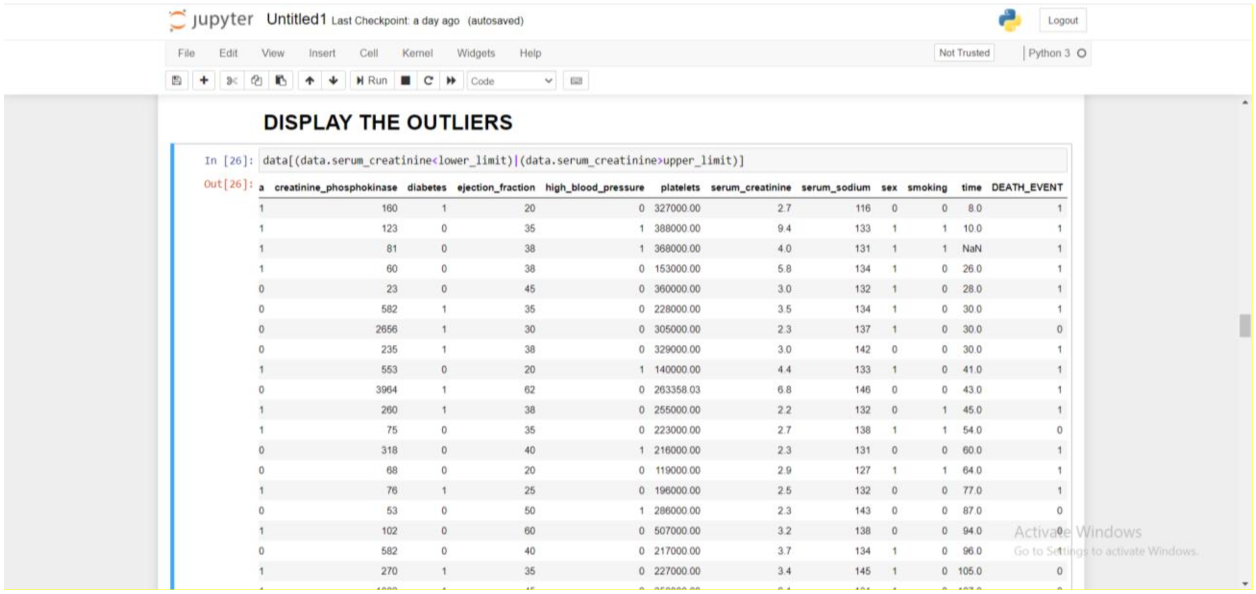
4.4 OUTLIER TREATMENT


jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
2 4 6 8
serum_creatinine

CALCULATION OF IQR

```
In [23]: Q1=data.serum_creatinine.quantile(0.25)
         Q3=data.serum_creatinine.quantile(0.75)
         Q1,Q3

Out[23]: (0.9, 1.4)

In [24]: IQR = Q3-Q1
         IQR

Out[24]: 0.4999999999999999

In [25]: lower_limit = Q1-1.5*IQR
         upper_limit = Q3+1.5*IQR
         lower_limit,upper_limit

Out[25]: (0.15000000000000024, 2.1499999999999995)
```

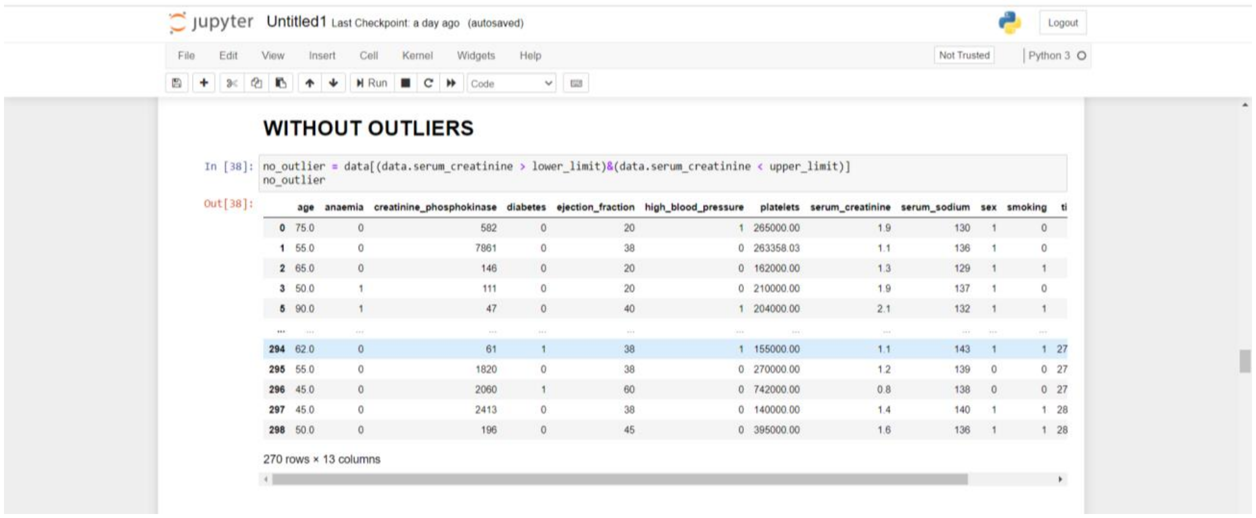
IQR means **INTERQURATILE RANGE**
Here we calculate the IQR range for serum_creatinine, calculate **25% & 75%,** the calculate the **IQR** range.
By using formula we find the **upper and lower limit.**


jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.

DISPLAY THE OUTLIERS

```
In [26]: data[(data.serum_creatinine<lower_limit)|(data.serum_creatinine>upper_limit)]

Out[26]:
a  creatinine_phosphokinase  diabetes  ejection_fraction  high_blood_pressure   platelets  serum_creatinine  serum_sodium  sex  smoking   time  DEATH_EVENT
1                       160         1                 20                    0   327000.00               2.7           116    0        0    8.0            1
1                       123         0                 35                    1   388000.00               9.4           133    1        1   10.0            1
1                        81         0                 38                    1   368000.00               4.0           131    1        1    NaN            1
1                        60         0                 38                    0   153000.00               5.8           134    1        0   26.0            1
0                        23         0                 45                    0   360000.00               3.0           132    1        0   28.0            1
0                       582         1                 35                    0   228000.00               3.5           134    1        0   30.0            1
0                      2656         1                 30                    0   305000.00               2.3           137    1        0   30.0            0
0                       235         1                 38                    0   329000.00               3.0           142    0        0   30.0            1
1                       553         0                 20                    1   140000.00               4.4           133    1        0   41.0            1
0                      3964         1                 62                    0   263358.03               6.8           146    0        0   43.0            1
1                       260         1                 38                    0   255000.00               2.2           132    0        1   45.0            1
1                        75         0                 35                    0   223000.00               2.7           138    1        1   54.0            0
0                       318         0                 40                    1   216000.00               2.3           131    0        0   60.0            1
0                        68         0                 20                    0   119000.00               2.9           127    1        1   64.0            1
1                        76         1                 25                    0   196000.00               2.5           132    0        0   77.0            1
0                        53         0                 50                    1   286000.00               2.3           143    0        0   87.0            0
1                       102         0                 60                    0   507000.00               3.2           138    0        0   94.0            0
0                       582         0                 40                    0   217000.00               3.7           134    1        0   96.0            1
1                       270         1                 35                    0   227000.00               3.4           145    1        0  105.0            0
```

Here we **display the outliers** only in the datset.
We can find that in the column serum_creatinine the **values are out of range**



WITHOUT OUTLIERS

```
In [38]: no_outlier = data[(data.serum_creatinine > lower_limit)&(data.serum_creatinine < upper_limit)]
         no_outlier
```

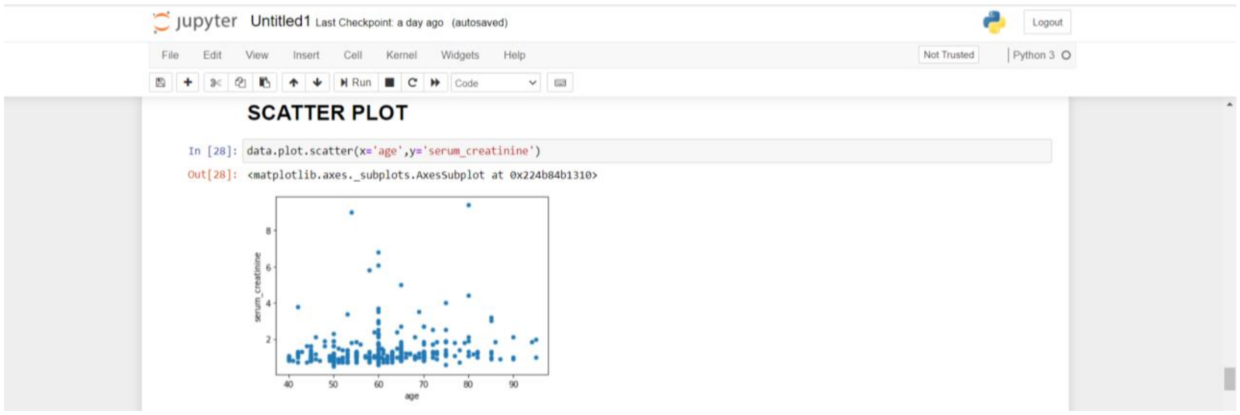
Out[38]:

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking	ti
0	75.0	0	582	0	20	1	265000.00	1.9	130	1	0	
1	55.0	0	7861	0	38	0	263358.03	1.1	136	1	0	
2	65.0	0	146	0	20	0	162000.00	1.3	129	1	1	
3	50.0	1	111	0	20	0	210000.00	1.9	137	1	0	
5	90.0	1	47	0	40	1	204000.00	2.1	132	1	1	
...	...	...	...	...	...	...	...	...	...	...	...	
294	62.0	0	61	1	38	1	155000.00	1.1	143	1	1	27
295	55.0	0	1820	0	38	0	270000.00	1.2	139	0	0	27
296	45.0	0	2060	1	60	0	742000.00	0.8	138	0	0	27
297	45.0	0	2413	0	38	0	140000.00	1.4	140	1	1	28
298	50.0	0	196	0	45	0	395000.00	1.6	136	1	1	28

270 rows × 13 columns

Here we can display the datasets **by removing the outliers.**
We can find that in the column serum_creatinine the values **are in between the IQR range**.

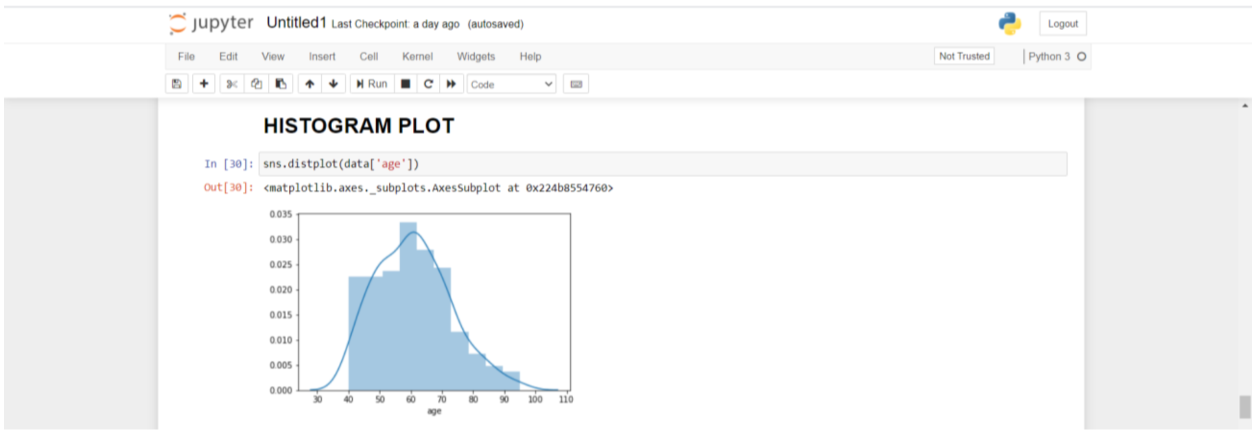
4.5 DATA VISUALIZATION



SCATTER PLOT

```
In [28]: data.plot.scatter(x='age',y='serum_creatinine')
Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x224b84b1310>
```

Here we compare age with serum_creatinine,.
By using scatter plot we came to know that inbetween the **age of 50 to 70** having more number of people’s is **less than 2** amount of serum_creatinine, less number of people’s in the **6 to 7 amount** od serum_creatine.

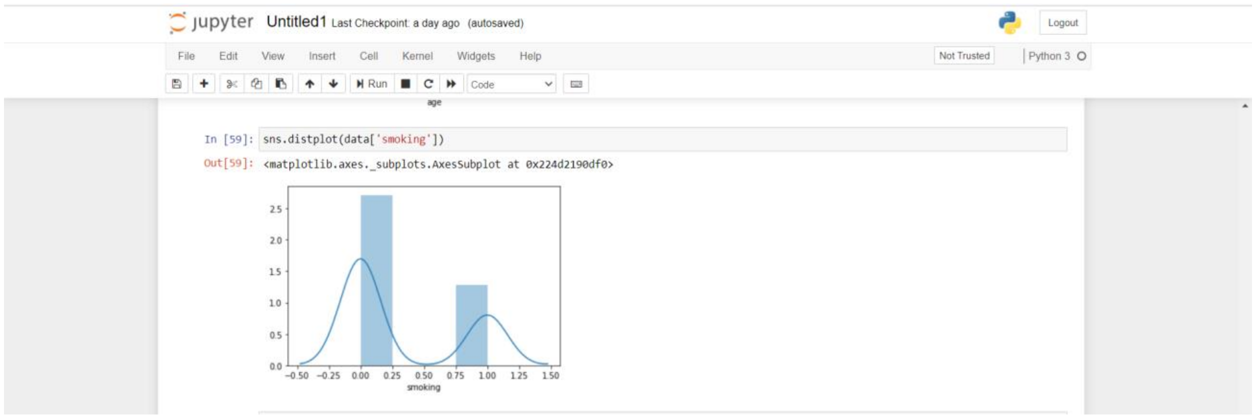

jupyter
Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
age

HISTOGRAM PLOT

```
In [30]: sns.distplot(data['age'])
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x224b8554760>
```

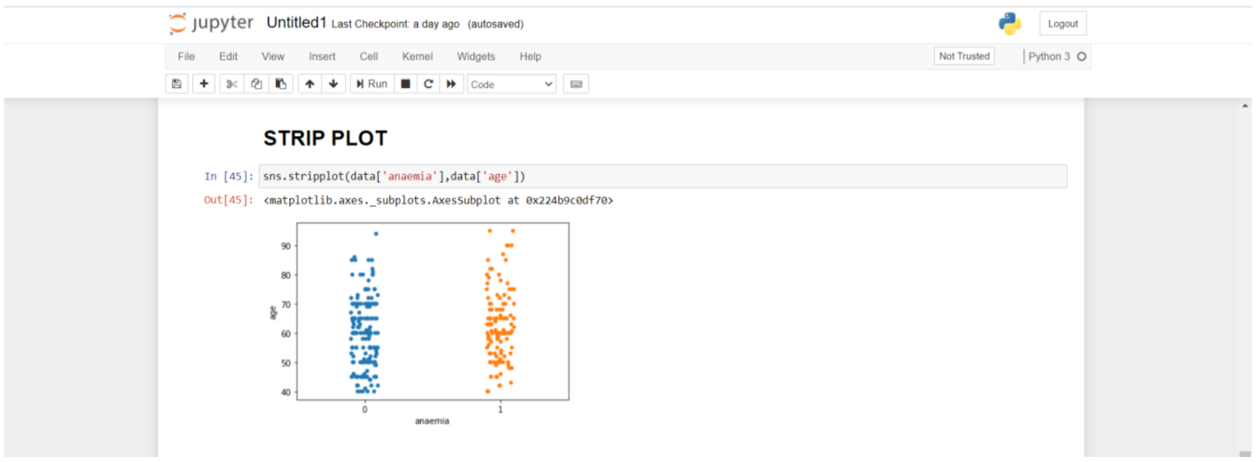
By using histogram plot, we know that level of increasing.
In this we analysis that,

- ✓ **22 percent** of people available in the **age of 40**
- ✓ **23 percent** of people available in the **age of 50**
- ✓ **24 percent** of people available in the **age of 55**
- ✓ **34 percent** of people available in the **age of 60**
- ✓ **26 percent** of people available in the **age of 62**
- ✓ **25 percent** of people available in the **age of 69**
- ✓ **11 percent** of people available in the **age of 71**
- ✓ **8 percent** of people available in the **age of 77**
- ✓ **5 percent** of people available in the **age of 82**
- ✓ **3 percent** of people available in the **age of 89**
- ✓ **0 percent** of people available in the **age of 93**


jupyter
Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
smoking

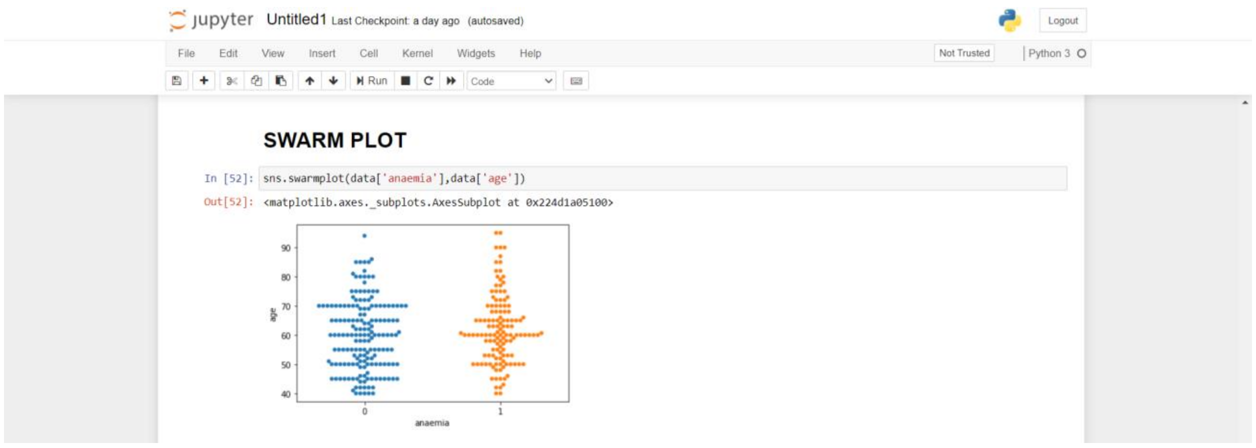
```
In [59]: sns.distplot(data['smoking'])
Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x224d2190df0>
```

Using histogram plot for categorical data, we came like this.
In this we done for smoking, **28 percent** of people **didn’t smoke**, only **13 percent** of people **smoke.**


jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
STRIP PLOT
age
anaemia

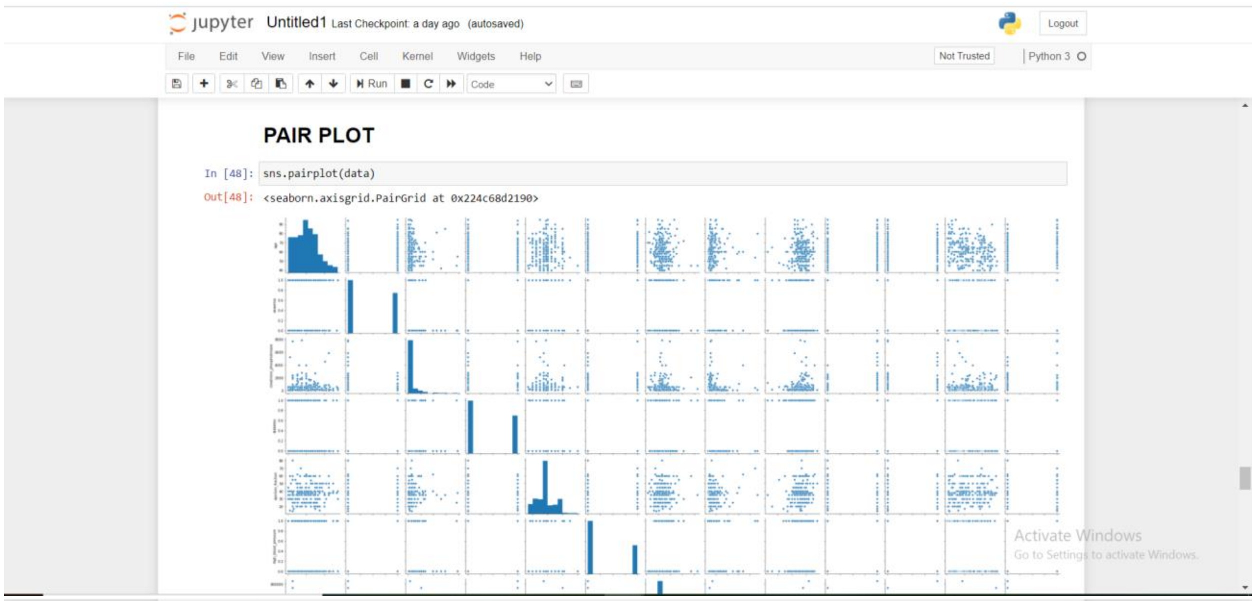
```
In [45]: sns.stripplot(data['anaemia'],data['age'])
Out[45]: <matplotlib.axes._subplots.AxesSubplot at 0x224b9c0df70>
```

Strip plot is one of the plot , this plot is specially used to analysis a categorical data Here we compare anaemia with age. We came to know that people in the age **between 60 to 70** have **affected** in anemia a lot. People **less than 40 age** have **not affected** as much.


jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
SWARM PLOT
age
anaemia

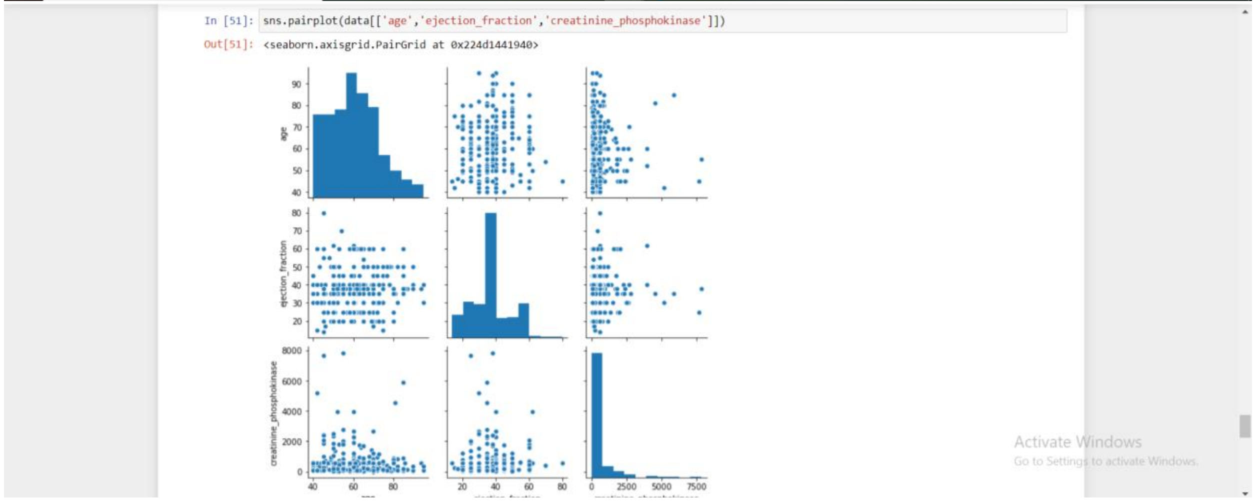
```
In [52]: sns.swarmplot(data['anaemia'],data['age'])
Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x224d1a05100>
```

Swarm plot is also for categorical data, as like a strip plot. But here the plots are in the **distributed manner**.


jupyter Untitled1 Last Checkpoint: a day ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
Activate Windows
Go to Settings to activate Windows.

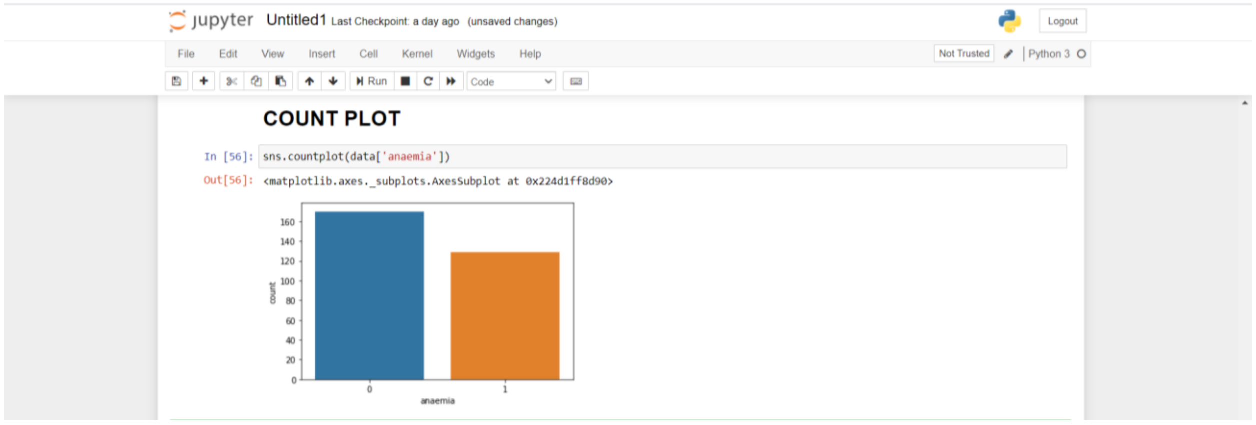
PAIR PLOT

```
In [48]: sns.pairplot(data)
Out[48]: <seaborn.axisgrid.PairGrid at 0x224c68d2190>
```



```
In [51]: sns.pairplot(data[['age','ejection_fraction','creatinine_phosphokinase']])
Out[51]: <seaborn.axisgrid.PairGrid at 0x224d1441940>
```

Pair plot


jupyter Untitled1 Last Checkpoint: a day ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
count
anaemia
0
1

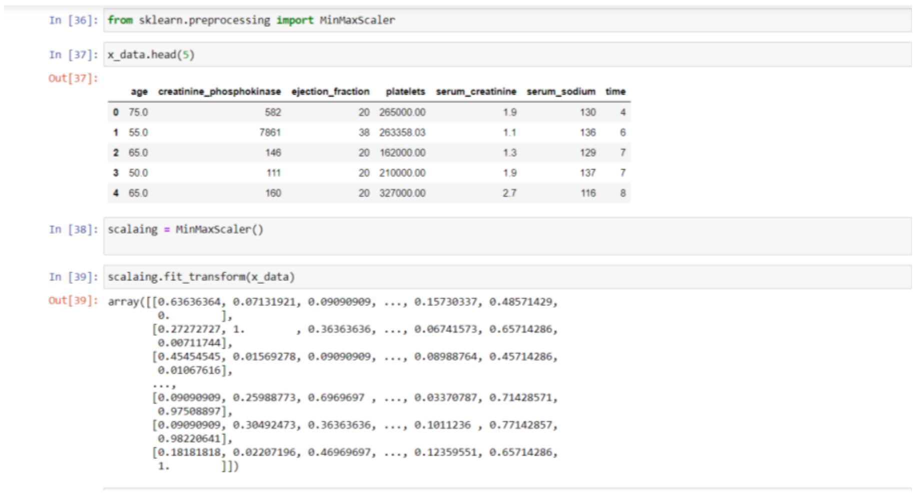
COUNT PLOT

```
In [56]: sns.countplot(data['anaemia'])
Out[56]: <matplotlib.axes._subplots.AxesSubplot at 0x224d1ff8d90>
```

Count plot is used to count number of peoples.
In this we analyze that **160 people's not affected** by anemia, **120 people's affected** by anemia.

4.6 VARIABLE TRANSFORMATION

NORMALISATION:

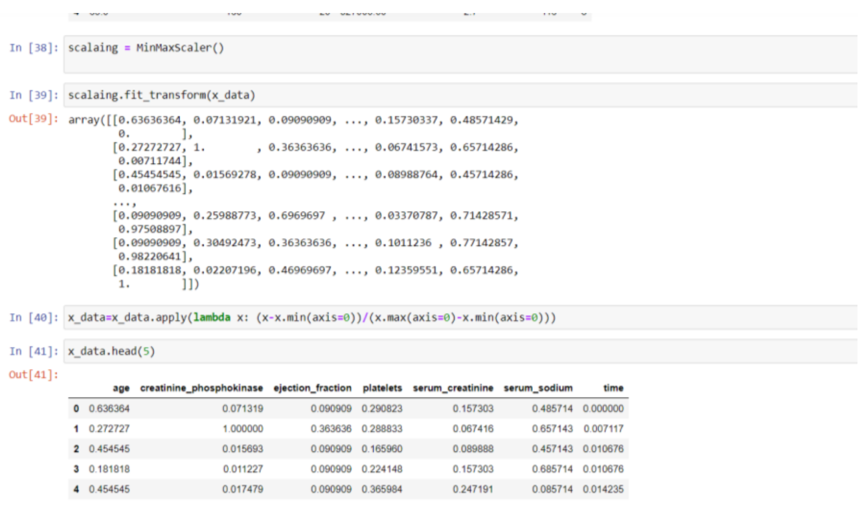


```
In [36]: from sklearn.preprocessing import MinMaxScaler

In [37]: x_data.head(5)
Out[37]:
```

	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium	time
0	75.0	582	20	265000.00	1.9	130	4
1	55.0	7861	38	263358.03	1.1	136	6
2	65.0	146	20	162000.00	1.3	129	7
3	50.0	111	20	210000.00	1.9	137	7
4	65.0	160	20	327000.00	2.7	116	8

```
In [38]: scalaing = MinMaxScaler()

In [39]: scalaing.fit_transform(x_data)
Out[39]: array([[0.63636364, 0.07131921, 0.09090909, ..., 0.15730337, 0.48571429,
                 0.        ],
                [0.27272727, 1.        , 0.36363636, ..., 0.06741573, 0.65714286,
                 0.00711744],
                [0.45454545, 0.01569278, 0.09090909, ..., 0.08988764, 0.45714286,
                 0.01067616],
                ...,
                [0.09090909, 0.25988773, 0.6969697 , ..., 0.03370787, 0.71428571,
                 0.97508897],
                [0.09090909, 0.30492473, 0.36363636, ..., 0.1011236 , 0.77142857,
                 0.98220641],
                [0.18181818, 0.02207196, 0.46969697, ..., 0.12359551, 0.65714286,
                 1.        ]])
```

We are using minmax sacalar to normalise the data between 0 -1.



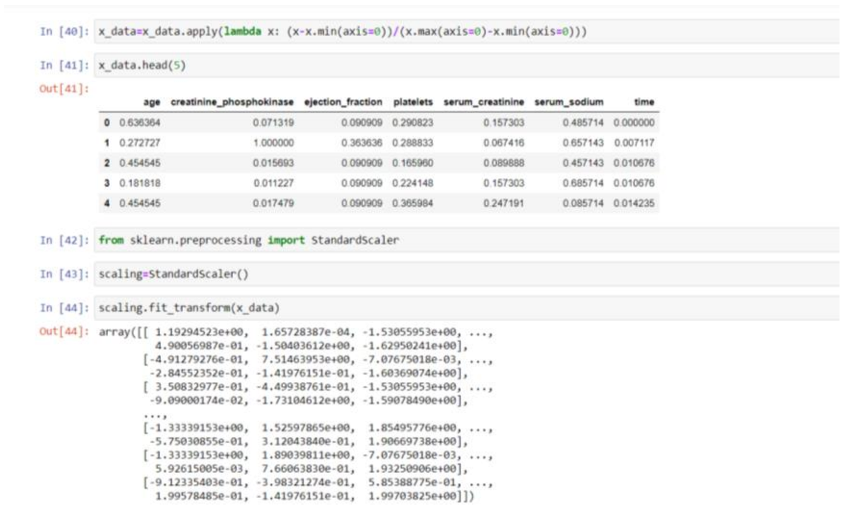
```
In [38]: scalaing = MinMaxScaler()

In [39]: scalaing.fit_transform(x_data)
Out[39]: array([[0.63636364, 0.07131921, 0.09090909, ..., 0.15730337, 0.48571429,
                 0.        ],
                [0.27272727, 1.        , 0.36363636, ..., 0.06741573, 0.65714286,
                 0.00711744],
                [0.45454545, 0.01569278, 0.09090909, ..., 0.08988764, 0.45714286,
                 0.01067616],
                ...,
                [0.09090909, 0.25988773, 0.6969697 , ..., 0.03370787, 0.71428571,
                 0.97508897],
                [0.09090909, 0.30492473, 0.36363636, ..., 0.1011236 , 0.77142857,
                 0.98220641],
                [0.18181818, 0.02207196, 0.46969697, ..., 0.12359551, 0.65714286,
                 1.        ]])
```

```
In [40]: x_data=x_data.apply(lambda x: (x-x.min(axis=0))/(x.max(axis=0)-x.min(axis=0)))

In [41]: x_data.head(5)
Out[41]:
```

	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium	time
0	0.636364	0.071319	0.090909	0.290823	0.157303	0.485714	0.000000
1	0.272727	1.000000	0.363636	0.288833	0.067416	0.657143	0.007117
2	0.454545	0.015693	0.090909	0.165960	0.089888	0.457143	0.010676
3	0.181818	0.011227	0.090909	0.224148	0.157303	0.685714	0.010676
4	0.454545	0.017479	0.090909	0.365984	0.247191	0.085714	0.014235

In this we are using lambda formula to normalise the data.

STANDARISATION:



```
In [40]: x_data=x_data.apply(lambda x: (x-x.min(axis=0))/(x.max(axis=0)-x.min(axis=0)))

In [41]: x_data.head(5)
Out[41]:
```

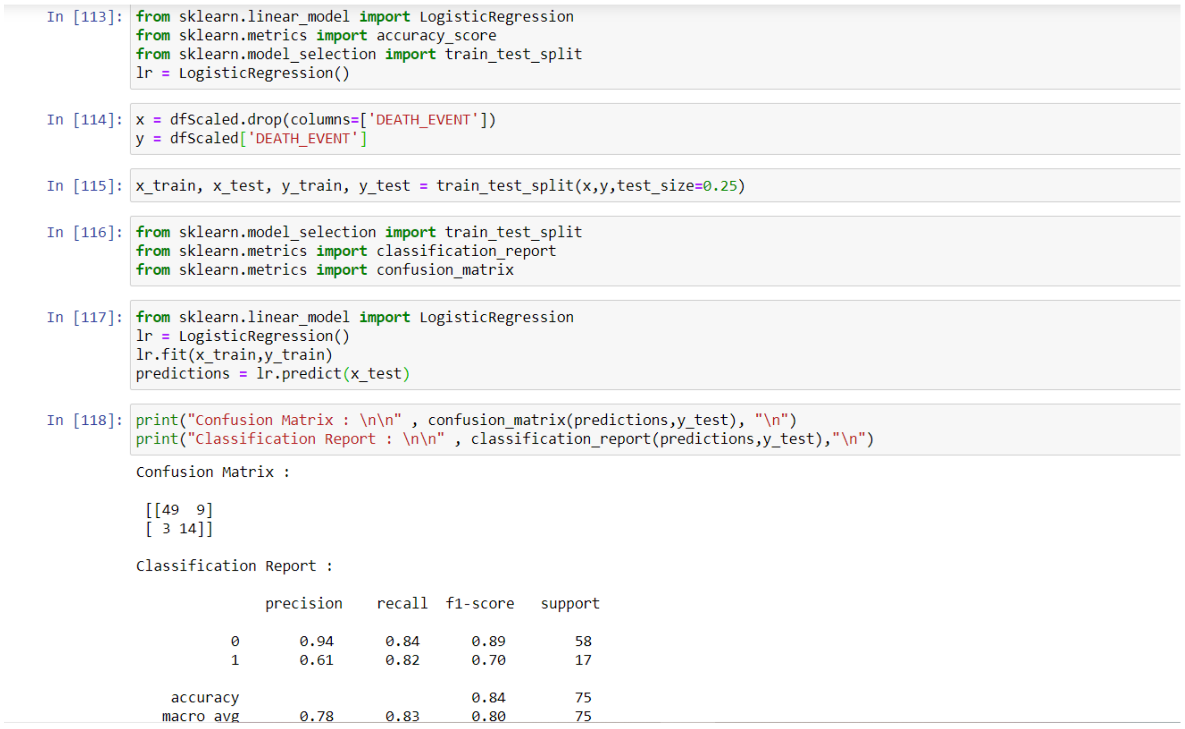
	age	creatinine_phosphokinase	ejection_fraction	platelets	serum_creatinine	serum_sodium	time
0	0.636364	0.071319	0.090909	0.290823	0.157303	0.485714	0.000000
1	0.272727	1.000000	0.363636	0.288833	0.067416	0.657143	0.007117
2	0.454545	0.015693	0.090909	0.165960	0.089888	0.457143	0.010676
3	0.181818	0.011227	0.090909	0.224148	0.157303	0.685714	0.010676
4	0.454545	0.017479	0.090909	0.365984	0.247191	0.085714	0.014235

```
In [42]: from sklearn.preprocessing import StandardScaler

In [43]: scaling=StandardScaler()

In [44]: scaling.fit_transform(x_data)
Out[44]: array([[ 1.19294523e+00,  1.65728387e-04, -1.53055953e+00, ...,
                  4.90056987e-01, -1.50403612e+00, -1.62950241e+00],
                [-4.91279276e-01,  7.51463953e+00, -7.07675018e-03, ...,
                 -2.84552352e-01, -1.41976151e-01, -1.60369074e+00],
                [ 3.50832977e-01, -4.49938761e-01, -1.53055953e+00, ...,
                 -9.09000174e-02, -1.73104612e+00, -1.59078490e+00],
                ...,
                [-1.33339153e+00,  1.52597865e+00,  1.85495776e+00, ...,
                 -5.75030855e-01,  3.12043840e-01,  1.90669738e+00],
                [-1.33339153e+00,  1.89039811e+00, -7.07675018e-03, ...,
                  5.92615005e-03,  7.66063830e-01,  1.93250906e+00],
                [-9.12335403e-01, -3.98321274e-01,  5.85388775e-01, ...,
                  1.99578485e-01, -1.41976151e-01,  1.99703825e+00]])
```

The dataset is standarised by having mean=0 and S.D=1

4.7 PREDICTION OF TARGET VARIABLE

CONFUSION MATRIX:



```
In [113]: from sklearn.linear_model import LogisticRegression
          from sklearn.metrics import accuracy_score
          from sklearn.model_selection import train_test_split
          lr = LogisticRegression()

In [114]: x = dfScaled.drop(columns=['DEATH_EVENT'])
          y = dfScaled['DEATH_EVENT']

In [115]: x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.25)

In [116]: from sklearn.model_selection import train_test_split
          from sklearn.metrics import classification_report
          from sklearn.metrics import confusion_matrix

In [117]: from sklearn.linear_model import LogisticRegression
          lr = LogisticRegression()
          lr.fit(x_train,y_train)
          predictions = lr.predict(x_test)

In [118]: print("Confusion Matrix : \n\n" , confusion_matrix(predictions,y_test), "\n")
          print("Classification Report : \n\n" , classification_report(predictions,y_test),"\n")

          Confusion Matrix :

           [[49  9]
           [ 3 14]]

          Classification Report :

                        precision    recall  f1-score   support

                     0       0.94      0.84      0.89        58
                     1       0.61      0.82      0.70        17

              accuracy                           0.84        75
             macro avg       0.78      0.83      0.80        75
```

It is used to test the data and predict each row in dataset.By using confusion matrix ,death event is compared with remaining attributes and gave 75 % accuracy.

5. CONCLUSION

By analysing the heart failure dataset the death rate is lower than the survival. The patients death is high only within 2 months.if the symptoms are more than 2 months then the chance to death is low.To date, there has not been a single trial targeting acute HF that has been shown to improve survival or readmission risk in these patients. Besides adhering to existing guidelines, newer approaches to managing these patients, both in terms of monitoring and developing novel therapeutic approaches, are desperately needed. These efforts are even more important now that readmission rates for HF is now a quality measure and hospitalization within 30 days of discharge will not be reimbursed to the hospitals. Thus, efforts to reduce HF hospitalization are clinically important and a fiscally relevant aim that represents a national priority