PROJECT PROPOSAL

1. **Title of the Project:** Heart Failure Prediction

2. **Brief on the project:**Heart disease is a huge difficulty to forecast in today's environment. There are several methods available that can forecast heart illness, but they are either expensive or unavailable to the general population, not to mention inefficient at calculating the likelihood of heart disease. Early identification of heart problems can help reduce mortality and future consequences. We have a lot of data floating around. We can analyse hidden patterns in data using various machine learning algorithms, which may provide useful insights for patients as well as the healthcare sector.

3. **Deliverables of the project:** The main objective of this project is:

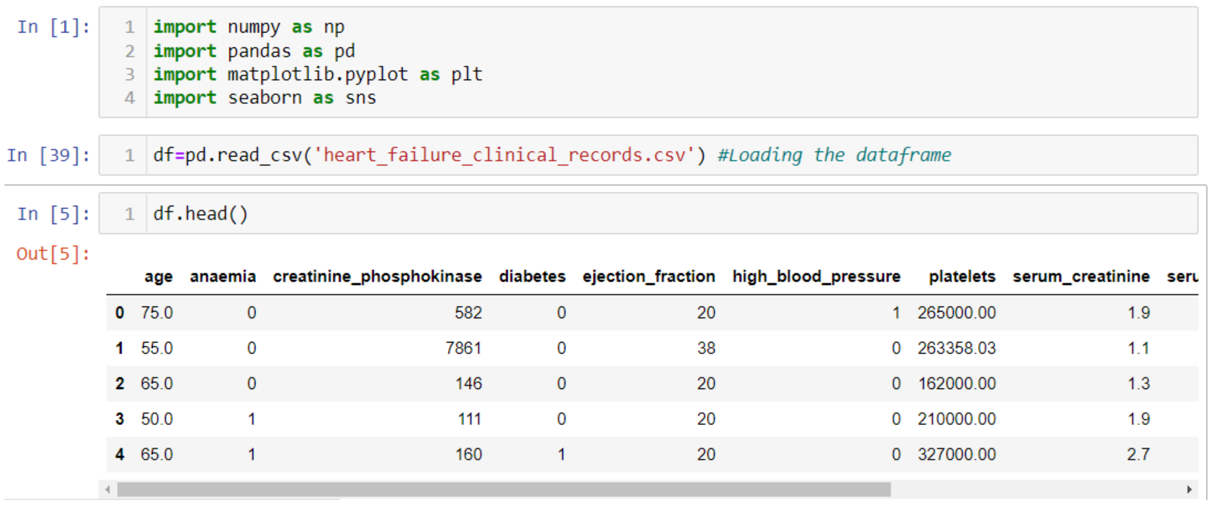
   - To develop a machine learning model to predict future possibility of heart disease by implementing Logistic Regression.
   - Another motive is to determine significant risk factors based on medical dataset which may lead to heart disease.
   - This project is also aims to understand several trends of the patients having heart disease to understand some other factors as well.

4. **Resources**

   - **Data set source:** The dataset was provided by of IIT, Roorkee.
   - **Software:** Python, Jupyter Notebook, Machine Learning libraries.
   - **References:**
     a) https://towardsdatascience.com/project-predicting-heart-disease-with-classification-machine-learning-algorithms-fd69e6fdc9d6.
     b) A mini project on Heart Disease Prediction, Kathmandu University, Department of Computer Science and engineering.

5. **Individual Details:** Prakhar Sahu, Email: prakharsahu2006@gmail.com,
   Phone No.: +91 8400668728

EXPLORATION AND PRE-PROCESSING OF THE DATASET:



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

In [39]: df=pd.read_csv('heart_failure_clinical_records.csv') #Loading the dataframe

In [5]:  df.head()
Out[5]:
```

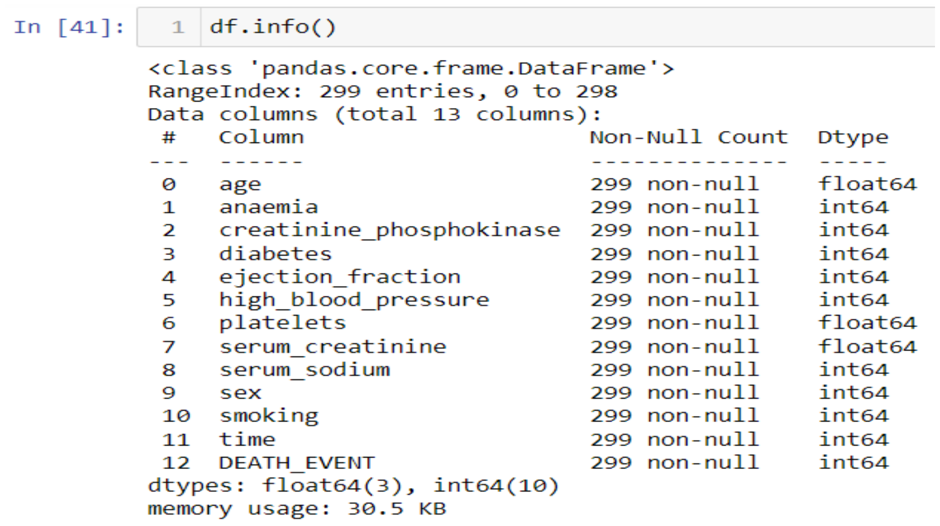
	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	seru
0	75.0	0	582	0	20	1	265000.00	1.9	
1	55.0	0	7861	0	38	0	263358.03	1.1	
2	65.0	0	146	0	20	0	162000.00	1.3	
3	50.0	1	111	0	20	0	210000.00	1.9	
4	65.0	1	160	1	20	0	327000.00	2.7	

Figure 1: *Dataset snapshot*

The dataset was was downloaded in the local drive for pre-processing in the Jupyter notebook using python libraries. The dataset have 13 attributes which includes: age, anaemia, creatinine_phosphokinase, diabetes, ejection_fraction, high_blood_pressure, platelets, serum_creatinine, serum_sodium, sex, smoking, time anddeath_event of over 299 patients. To pre-process the data, we have initially imported the following files:

- Numpy
- Pandas
- Matplotlib
- Seaborn.

The dataset was loaded into Jupiter notebook using pandas "pd.read" command. Since the dataset given was in CSV format, we have used "pd.read_csv" to load the dataset into a variable called "df".



```
In [41]: df.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                      299 non-null    int64  
 12  DEATH_EVENT               299 non-null    int64  
dtypes: float64(3), int64(10)
memory usage: 30.5 KB
```

Figure 2: *Dataset attributes*

Using “info()” function, the attributes of the dataset was checked for any null value, as the data shows, there are no null values in the dataset.

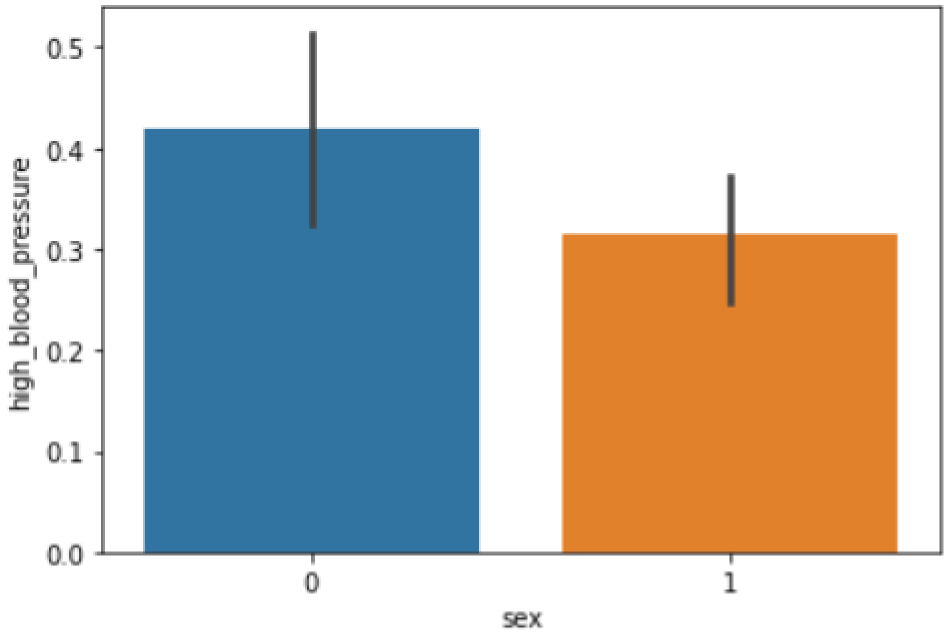
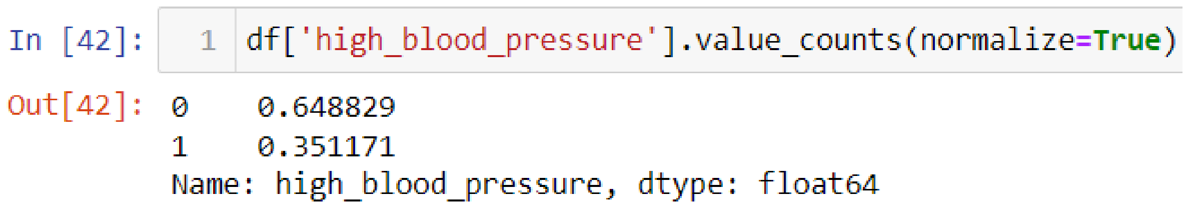


Figure 3: *High blood pressure representation between males and females*

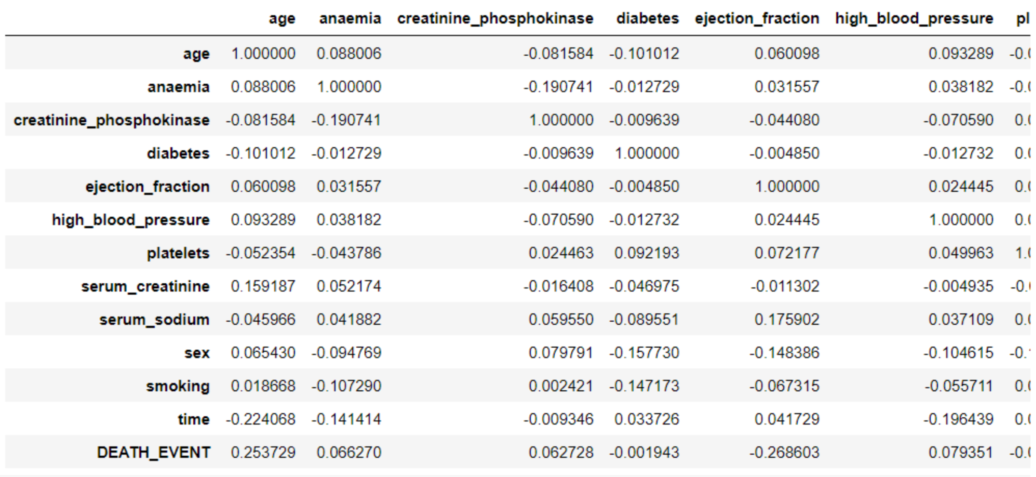


```
In [42]:   df['high_blood_pressure'].value_counts(normalize=True)

Out[42]: 0    0.648829
         1    0.351171
         Name: high_blood_pressure, dtype: float64
```

Figure 4: *High blood pressure ratio in percentage*

If we compare the data for High blood pressure, We can say that the overall percentage of males (represented by 0) is higher than females (represented by 1). There are around 65% males in comparison to 35% females having high blood pressure (see figure 4).



	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	pl
age	1.000000	0.088006	-0.081584	-0.101012	0.060098	0.093289	-0.(
anaemia	0.088006	1.000000	-0.190741	-0.012729	0.031557	0.038182	-0.(
creatinine_phosphokinase	-0.081584	-0.190741	1.000000	-0.009639	-0.044080	-0.070590	0.(
diabetes	-0.101012	-0.012729	-0.009639	1.000000	-0.004850	-0.012732	0.(
ejection_fraction	0.060098	0.031557	-0.044080	-0.004850	1.000000	0.024445	0.(
high_blood_pressure	0.093289	0.038182	-0.070590	-0.012732	0.024445	1.000000	0.(
platelets	-0.052354	-0.043786	0.024463	0.092193	0.072177	0.049963	1.(
serum_creatinine	0.159187	0.052174	-0.016408	-0.046975	-0.011302	-0.004935	-0.(
serum_sodium	-0.045966	0.041882	0.059550	-0.089551	0.175902	0.037109	0.(
sex	0.065430	-0.094769	0.079791	-0.157730	-0.148386	-0.104615	-0.
smoking	0.018668	-0.107290	0.002421	-0.147173	-0.067315	-0.055711	0.(
time	-0.224068	-0.141414	-0.009346	0.033726	0.041729	-0.196439	0.(
DEATH_EVENT	0.253729	0.066270	0.062728	-0.001943	-0.268603	0.079351	-0.(

Figure 5: *Correlation of attributes.*

Using correlation function of Numpy library, we have also checked the correlations of various attributes in the datasets (figure 5).

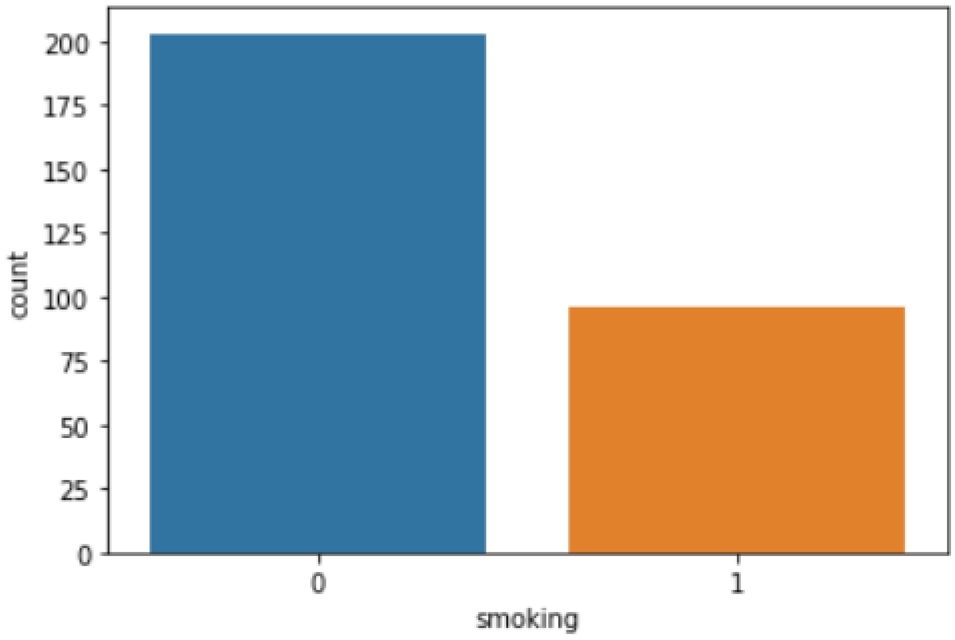


Figure 6:*Smoking habits comparison.*

The countplot function of pandas library shows that there are lots of male smokers in comparison to females. The bar represented with blue colour is male (approximate 200) while the orange one represents females (approximate 100 in figure 9).

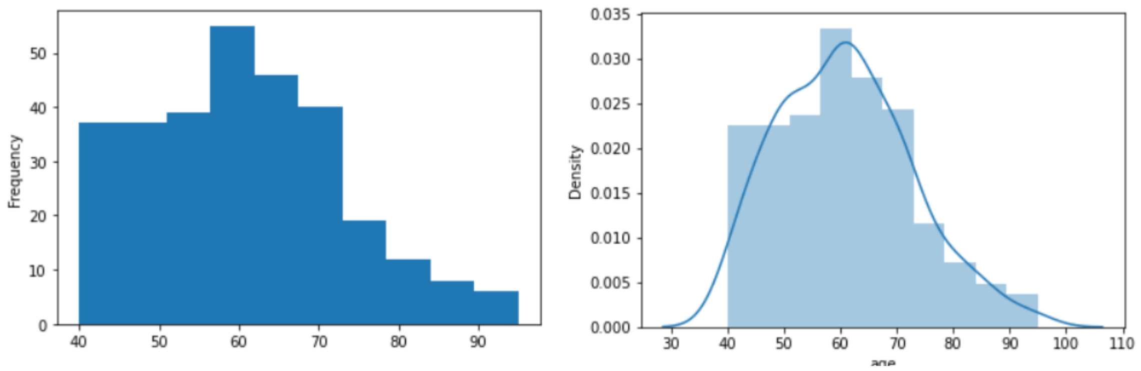
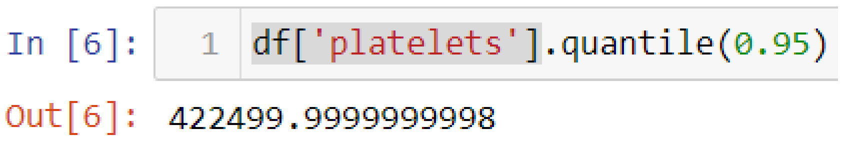


Figure 7:*Age frequency and skewness.*

Using the histogram function of Matplotlib, we have created an age histogram to represent which age bracket is higher. The results in figure 6 shows that patients of age between 60 to 70 years have higher impact on the dataset (see figure 7).

CLINICAL DATA COMPARISON:

Platelets Count: Using the quantile function of Numpy library, we have fetched the data of platelets count of 95% population (see figure 8).

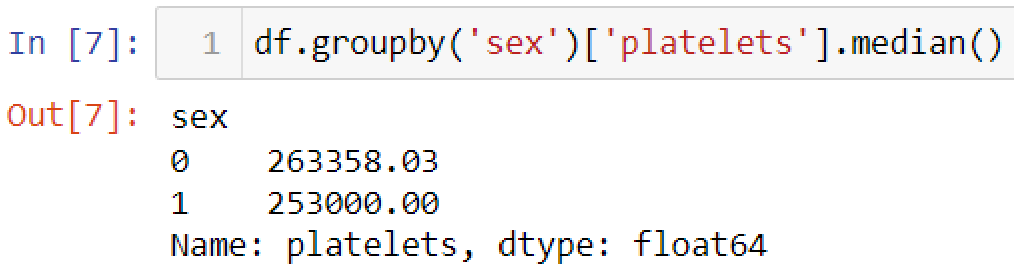


```
In [6]:   1  df['platelets'].quantile(0.95)

Out[6]:  422499.9999999998
```

Figure 8:*Platelets count of 95% population.*

As the figure 8 shows, The platelet count of 95% population falls under 42000.



```
In [7]:   1  df.groupby('sex')['platelets'].median()

Out[7]:  sex
         0     263358.03
         1     253000.00
         Name: platelets, dtype: float64
```

Figure 9:*Median platelets count of males and females.*

Figure 9 shows that the median platelets count of male patients is 263358 and for females its 253000.

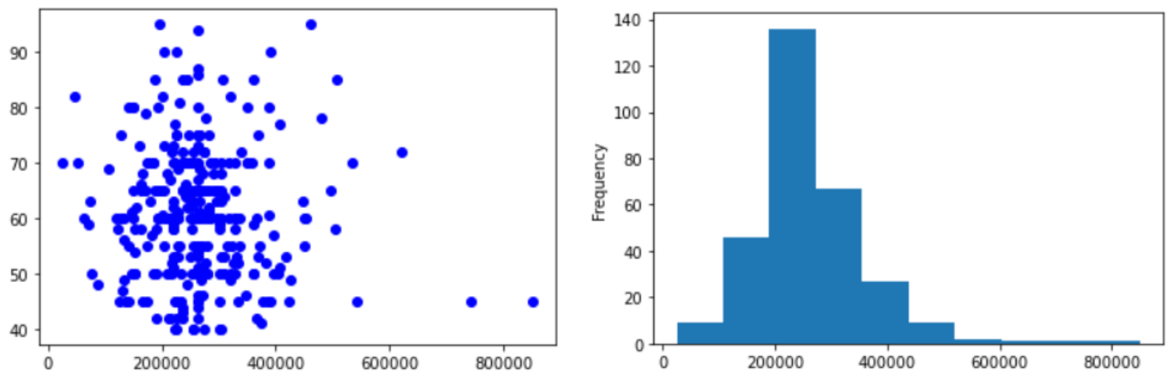


Figure 10:*Scatterplot of platelets count and frequency.*

We have also created a scatterplot to see the platelets count variation over ages. In figure 10, it is clearly visible the platelets count from 40 years to 70 years stays under 40000 while there is a significant variation in platelets count after that. Frequency histogram of platelets counts is also shown.

Anaemia: Using the countplot function of Seaborn library, we have created a countplot to check the impressions of Anaemia among genders (see figure 11).

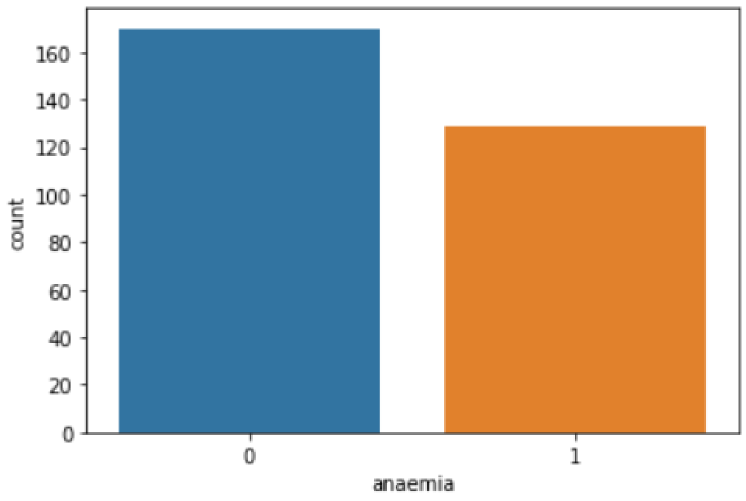


Figure 11:*Countplot of Anaemia.*

The data provided shows that there are more than 160 males affected with Anaemia than while females are limited between 120 to 140.

Diabetes: Figure 12 depicts the condition of diabetes among male and femalespatients. A counplot was created with "hue" function of seaborn to see the difference between diabetic and non-diabetic patients.

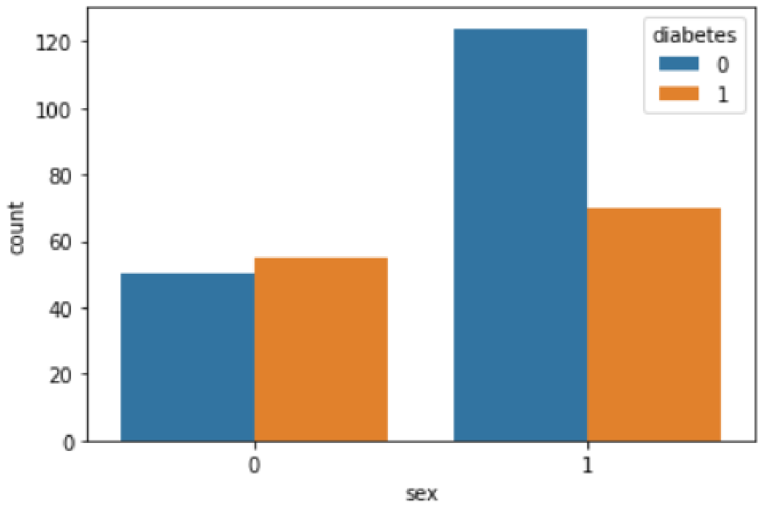
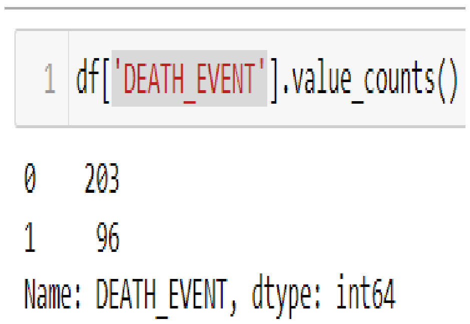


Figure 12:*Diabetes countplot.*

Using the above countplot the following readings were observed:

- *Non-diabetic males: 40-50.*
- *Diabetic males: 50-60.*
- *Non-diabetic females: 120-125.*
- *Diabetic females: 60-70.*

Death Events: Using "value_counts()" function of pandas library, we have fetched the values of Death Event attribute to determine overall deaths according to the dataset given. The observations are as follows:



```
df['DEATH_EVENT'].value_counts()

0    203
1     96
Name: DEATH_EVENT, dtype: int64
```

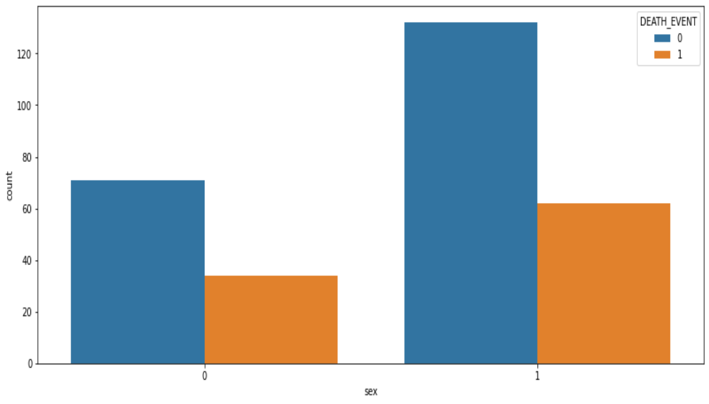
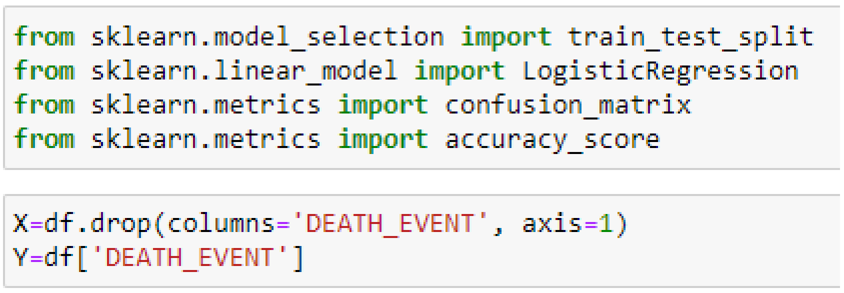


Figure 13: *Overall death events calculation using Pandas value_counts function.*

There are around 96 deaths (represented by 1) events recorded as per the dataset while 203 patients were alive (represented by 0). The bargraph in figure 13 shows that there more overall deaths of females are higher than males.

MODEL CREATION:

<u>USING LOGISTIC REGRESSION:</u> The model is trained and tested using Logistic Regression Algorithm to predict if user has a risk of heart disease or not.

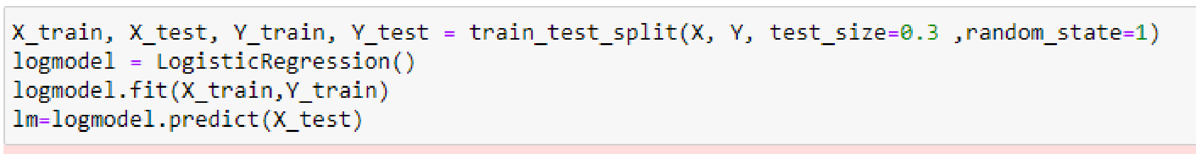


```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

```
X=df.drop(columns='DEATH_EVENT', axis=1)
Y=df['DEATH_EVENT']
```

Figure 14: *Importing necessary sklearn library files for Logistic Regression.*

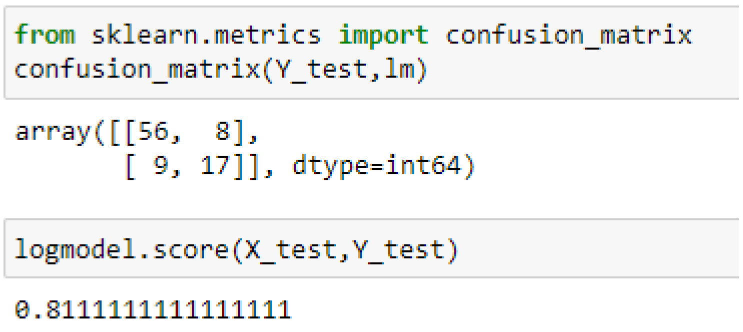
For Model creation for Heart Disease prediction using Logistic Regression, we have imported above files and bifurcated the dataframe into two parts X and Y. Variable X has 12 attributes and Y has only one attribute "Death_Event", used as a dependant variable (see figure 14).



```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3 ,random_state=1)
logmodel = LogisticRegression()
logmodel.fit(X_train,Y_train)
lm=logmodel.predict(X_test)
```

Figure 15: *Training and testing the dataset using Train Test Split Function.*

The dataset was trained and tested using Train Test Split function of Sklearn libraries. The test size was 0.3% and the train size was 0.7% (see figure 15).



```
from sklearn.metrics import confusion_matrix
confusion_matrix(Y_test,lm)
```

```
array([[56,  8],
       [ 9, 17]], dtype=int64)
```

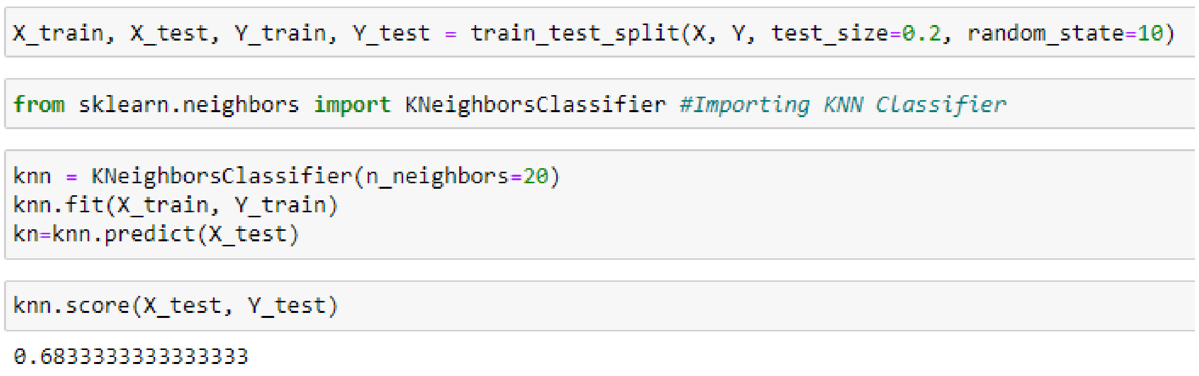
```
logmodel.score(X_test,Y_test)
```

```
0.8111111111111111
```

Figure 16: *Training and testing the dataset using Train Test Split Function.*

Results: A confusion matrix along with accuracy score was checked. The accuracy of the model using Logistic Regressing was 81% (see figure 16).

<u>USING K NEAREST NEIGHBOUR:</u>



```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=10)
```

```
from sklearn.neighbors import KNeighborsClassifier #Importing KNN Classifier
```

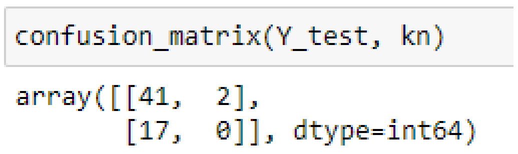
```
knn = KNeighborsClassifier(n_neighbors=20)
knn.fit(X_train, Y_train)
kn=knn.predict(X_test)
```

```
knn.score(X_test, Y_test)
```

```
0.6833333333333333
```

Figure 17: *Training and testing the dataset using Train Test Split Function using KNN.*

To train the model using K Nearest Neighbour was same procedure was followed except changes in few parts like in testing part we have using 20% of the size with 20 nearest neighbours. The accuracy of the model was 68% (see figure 17). A confusion matrix was also plotted (see figure 18).



```
confusion_matrix(Y_test, kn)
```

```
array([[41,  2],
       [17,  0]], dtype=int64)
```

Figure 18: *Confusion matrix using KNN.*

CONCLUSION: The early diagnosis of cardiovascular illnesses can help high-risk individuals make lifestyle adjustments and, as a result, prevent problems, which can be a significant milestone in the area of medicine. Logistic Regression and KNN were employed as models. In order to improve it, we can train models to forecast the sorts of cardiovascular problems and provide advice to users, as well as employ more advanced algorithms..