The dataset used in the given project has been obtained from Kaggle  
named as Heart failure prediction dataset by Larxel.The dataset can be  
accessed by clicking below.  
[https://www.kaggle.com/andrewmvd/heart-failure-clinical-data](https://www.kaggle.com/andrewmvd/heart-failure-clinical-data" \t "https://mail.google.com/mail/u/0/" \l "sent/_blank)  
  
The models that have been considered are Logistic regression, Support  
vector classifier, K-nearest neighbours classifier, decision tree  
classifier and random forest classifier.  
The best machine learning model has been obtained by comparing the  
accuracy scores of the model as well as the precision and recall  
values.  
The model has been created using Python programming language and has  
been attached in the mail. The output and observations are in the  
attached word document.  
The dataset has also been attached as an excel file.

**Aim:** To build a machine learning classification model using Python for predicting the whether a person having a heart failure will survive or not based on various factors like age, gender, follow up period (time) , weather a person has anemia or diabetes or not etc.

**Procedure:**

Here we take the death event i.e the death of a patient in the follow up period as the dependent variable and we take the potential independent variables as

* **Age**
* **Whether the patient has anemia (binary 1 if yes)**
* **Whether the patient has diabetes (binary 1 if yes)**
* **Whether the patient smokes**
* **Gender (binary 1 if female)**
* **Levels of creatinine phosphokinase**
* **Levels of ejection fraction**
* **Levels of platelets**
* **Levels of serum creatinine**
* **Levels of serum sodium**
* **follow up period (time)**

Hence we first check the correlation heat map to see which of the following potential independent variables have significant (both negative and positive) correlations with the dependent variable.

Then we create a training set and testing set from the data provided to us.

Based on the training set we create 5 different machine learning models for prediction of death event and compare their accuracy with respect to the test set given to us.

**Input and code:**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import GridSearchCV

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

df=pd.read\_csv('C:/Users/pishu/Downloads/727551\_1263738\_bundle\_archive/heart\_failure\_clinical\_records\_dataset.csv')

print(df.head())

print(df.isnull().sum(),'\n\n')

corr=df.corr()

sns.heatmap(corr,xticklabels=corr.columns,yticklabels=corr.columns,cmap='coolwarm')

plt.show()

# Hence we know check which of the variables have absolute value of correlation greater than 0.15 with the death event variable

print(corr[abs(corr['DEATH\_EVENT'])>0.15]['DEATH\_EVENT'])

x=df[['age','ejection\_fraction','serum\_creatinine','time','serum\_sodium']]

y=df['DEATH\_EVENT']

accuracy={}

# we de\ivide the given dataset into training set and test set

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=2698)

# 1) Logistic Regression

logreg=LogisticRegression()

logreg.fit(x\_train,y\_train)

logreg\_pred=logreg.predict(x\_test)

accuracy['Logistic Regression']=accuracy\_score(y\_test,logreg\_pred)

print('\n',classification\_report(y\_test,logreg\_pred))

# 2) Support Vector Classifier

svm=SVC()

svm.fit(x\_train,y\_train)

svm\_pred=svm.predict(x\_test)

accuracy['SVC']=accuracy\_score(y\_test,svm\_pred)

print('\n',classification\_report(y\_test,svm\_pred))

# 3) K nearest neighbors with hyperparameter testing of the number of nearest neighbors to be considered

n\_neighbours= list(range(1,10))

knn=KNeighborsClassifier()

params=dict(n\_neighbors=n\_neighbours)

knn\_2=GridSearchCV(knn,params,cv=5)

knn\_2.fit(x\_train,y\_train)

knn\_2\_pred=knn\_2.predict(x\_test)

accuracy['K Nearest Neighbors']=accuracy\_score(y\_test,knn\_2\_pred)

print('\n',classification\_report(y\_test,knn\_2\_pred))

# 4) Decision Tree Classifier

dt=DecisionTreeClassifier(max\_leaf\_nodes=10,random\_state=30,criterion='entropy')

dt.fit(x\_train,y\_train)

dt\_pred=dt.predict(x\_test)

accuracy['Decision Tree Classifier']=accuracy\_score(y\_test,dt\_pred)

print('\n',classification\_report(y\_test,dt\_pred))

#5) Random Forest Classifier with hyperparameter testing for the max\_depth

max\_depth=list(range(1,50))

params=dict(max\_depth=max\_depth)

fr\_1=RandomForestClassifier(max\_features=0.5,random\_state=1)

fr=GridSearchCV(fr\_1,params,cv=5)

fr.fit(x\_train,y\_train)

fr\_pred=fr.predict(x\_test)

accuracy['Random Forest Classifier']=accuracy\_score(y\_test,fr\_pred)

print('\n',classification\_report(y\_test,fr\_pred))

print(accuracy)

**Output:**

age anaemia creatinine\_phosphokinase ... smoking time DEATH\_EVENT

0 75.0 0 582 ... 0 4 1

1 55.0 0 7861 ... 0 6 1

2 65.0 0 146 ... 1 7 1

3 50.0 1 111 ... 0 7 1

4 65.0 1 160 ... 0 8 1

[5 rows x 13 columns]

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

age 0.253729

ejection\_fraction -0.268603

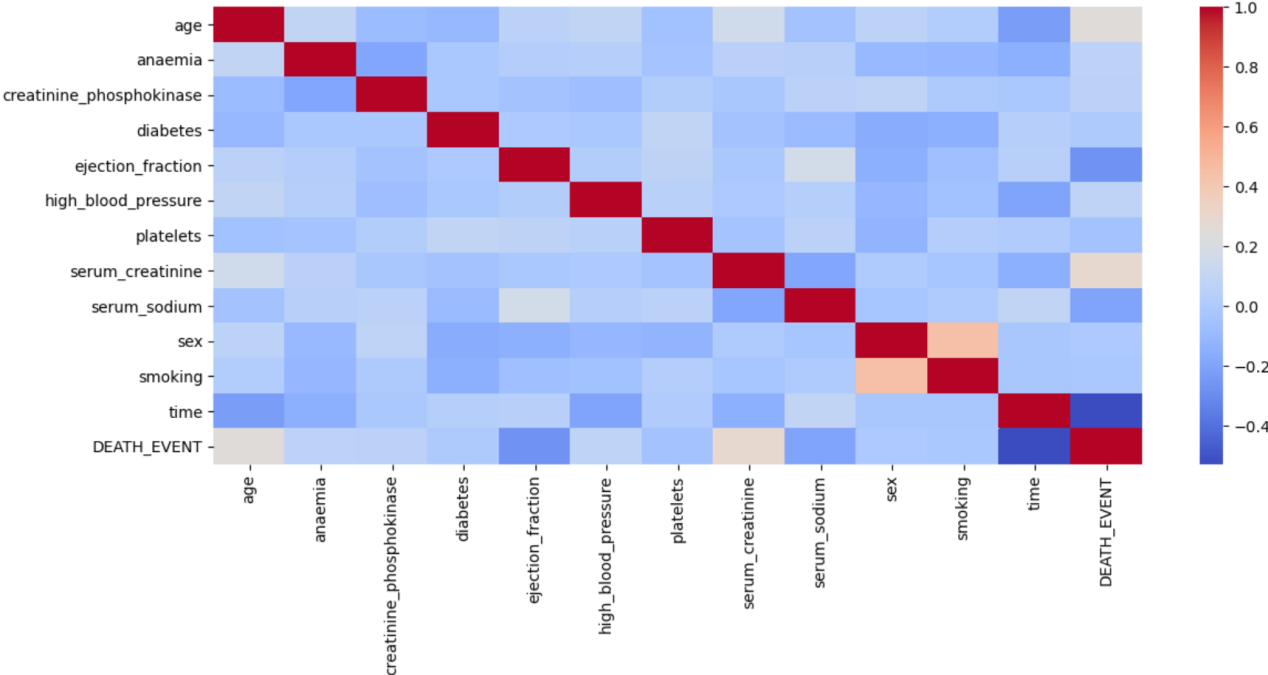
serum\_creatinine 0.294278

serum\_sodium -0.195204

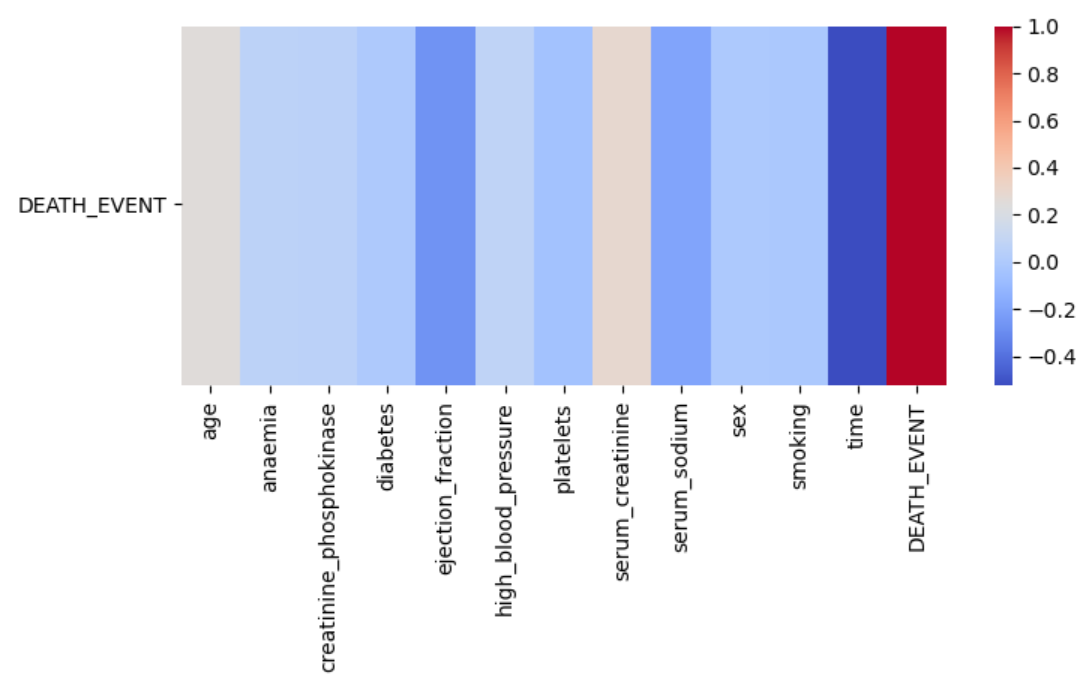
time -0.526964

DEATH\_EVENT 1.000000

Name: DEATH\_EVENT, dtype: float64



**Zoomed in for the death event only -->**



precision recall f1-score support

0 0.92 0.96 0.94 47

1 0.82 0.69 0.75 13

accuracy 0.90 60

macro avg 0.87 0.82 0.84 60

weighted avg 0.90 0.90 0.90 60

precision recall f1-score support

0 0.92 0.98 0.95 47

1 0.90 0.69 0.78 13

accuracy 0.92 60

macro avg 0.91 0.84 0.87 60

weighted avg 0.92 0.92 0.91 60

precision recall f1-score support

0 0.94 0.96 0.95 47

1 0.83 0.77 0.80 13

accuracy 0.92 60

macro avg 0.89 0.86 0.87 60

weighted avg 0.91 0.92 0.92 60

precision recall f1-score support

0 0.94 0.98 0.96 47

1 0.91 0.77 0.83 13

accuracy 0.93 60

macro avg 0.92 0.87 0.90 60

weighted avg 0.93 0.93 0.93 60

precision recall f1-score support

0 0.94 0.98 0.96 47

1 0.91 0.77 0.83 13

accuracy 0.93 60

macro avg 0.92 0.87 0.90 60

weighted avg 0.93 0.93 0.93 60

{'Logistic Regression': 0.9, 'SVC': 0.9166666666666666, 'K Nearest Neighbors': 0.9166666666666666, 'Decision Tree Classifier': 0.9333333333333333, 'Random Forest Classifier': 0.9333333333333333}

>>>

**Observations:**

Here we can eisely see that there is significant correlation between death event with the variables age,ejection\_fraction , serum\_creatinine , time and serum\_sodium.

Henceforth we consider only these items in the prediction model that we intend to create.

Now when we create the models the we get the following accuracy

* **Logistic Regression - 0.9**
* **Support Vector Classifier- 0.916**
* **K Nearest Neighbour - 0.916**
* **Decision Tree classifier - 0.93**
* **Random Forest Classifier - 0.93**

Henceforth as Both Random Forest Classifier and Decision tree classifier give us the same accuracy we can use either of them for prediction.

Also as their precision(ability of the classifier not to label as positive a sample that is negative) and recall(ability of the classifier to find all the positive samples) values are equal hence both these machine learning models are equally good and intuitive.