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 a machine learning model can be of great help.

Dataset Features

1. Age = age of the patients
2. Anaemia - Decrease of red blood cells or hemoglobin
3. Creatinine_phosphokinase - Level of the CPK enzyme in the blood (mcg/L)
4. Diabetes - If the patient has diabetes
5. Ejection_fraction - Percentage of blood leaving the heart at each contraction
6. High_blood_pressure - If the patient has hypertension
7. Platelets - Platelets in the blood (kiloplatelets/mL)
8. Serum_creatinine - Level of serum creatinine in the blood (mg/dL)
9. Serum_sodium - Level of serum sodium in the blood (mEq/L)
10. Sex - Woman or man
11. Smoking - If the patient smokes or not
12. Time - Follow-up period (days)

Dependent Variable

DEATH_EVENT

- If the patient deceased during the follow-up period

Import Packages

In [1]:
```
%config Completer.use_jedi = False
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from collections import Counter
from scipy import stats
```

In [2]:
```
data = pd.read_csv('heart_failure_clinical_records_dataset.csv')
data.head()
```

Out[2]:

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

EDA

EDA consists of:

- Finding Missing Values.
- Check for discrete and continuous variables for easy visualization.
- Correlation Matrix/Heatmap for finding relationship between independent variables and dependent variable.
- Also the heatmap depicts the correlation between the feature sets so that one of the correlated features can be dropped.
- Using Countplot, find the distribution of each feature individually and also wrt dependent variable.
- Find the balance in the dependent variable, so that necessary steps can be taken.
- Outliers Correction and Distribution Graph for better understanding of data with dependent variable.

In [3]:
```
data.isnull().sum()
```

Out[3]:
```
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
```

Observations

- No missing values present.
- So let's see the analysis ahead.

In [4]:
```
for feature in data.columns:
    print(feature, ':', len(data[feature].unique()))
```

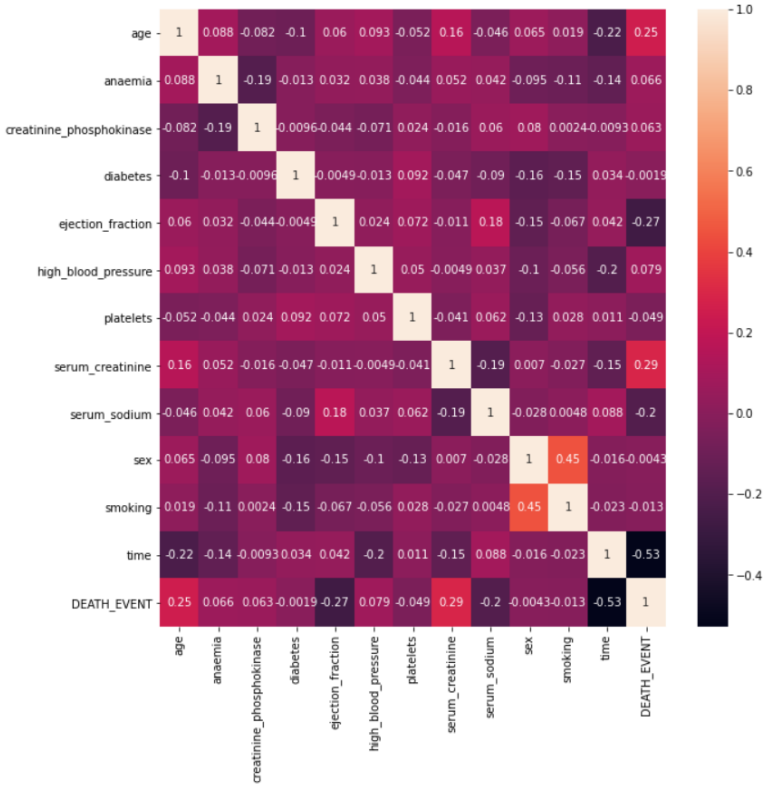
```
age : 47
anaemia : 2
creatinine_phosphokinase : 208
diabetes : 2
ejection_fraction : 17
high_blood_pressure : 2
platelets : 176
serum_creatinine : 40
serum_sodium : 27
sex : 2
smoking : 2
time : 148
DEATH_EVENT : 2
```

In [5]:
```
discrete_features, continuous_features = [], []
for feature in data.columns:
    if feature == 'DEATH_EVENT':
        label = ['DEATH_EVENT']
    elif len(data[feature].unique()) >= 10:
        continuous_features.append(feature)
    else:
        discrete_features.append(feature)

print('Discrete: ', discrete_features, '\n', 'Continuous', continuous_features)
```

```
Discrete:  ['anaemia', 'diabetes', 'high_blood_pressure', 'sex', 'smoking']
 Continuous ['age', 'creatinine_phosphokinase', 'ejection_fraction', 'platelets', 'serum_creatinine', 'serum_sodium', 'time']
```

In [6]:
```
correlation = data.corr()
plt.figure(figsize=(10, 10))
sns.heatmap(correlation, annot=True)
plt.show()
```

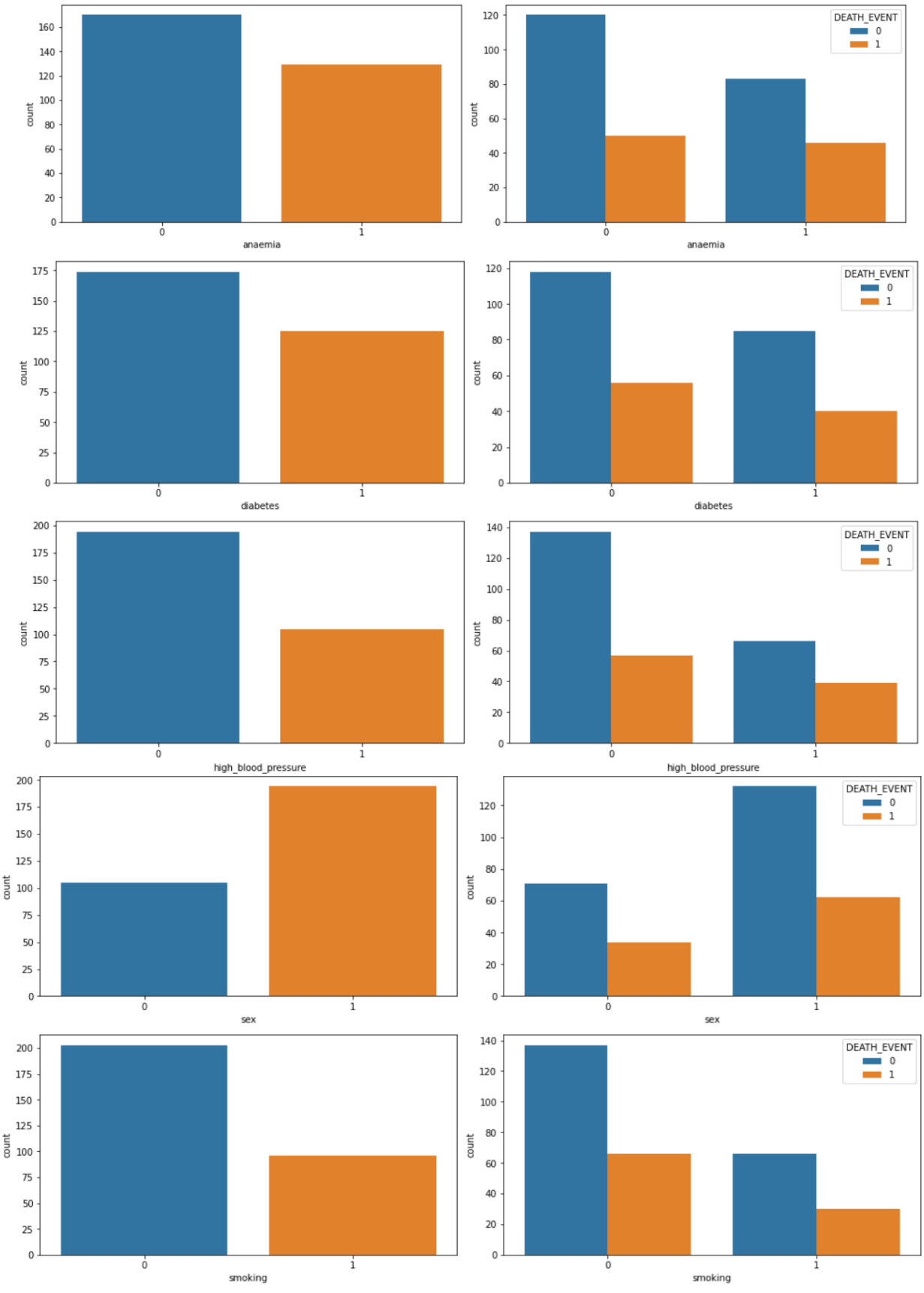


Observations

- There is nothing to conclude from discrete features correlation matrix.
- From the correlation matrix for continuous features, time is inversely correlated to death. Thus patients with less follow up time are prone to heart failure.
- Based on EDA, features such as **anaemia, diabetes, age, sex, smoking** are less contributing.

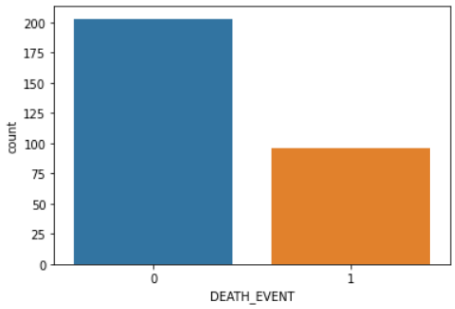
In [7]:
```
fig, ax = plt.subplots(len(discrete_features), 2, figsize=(14,20))

for i in range(len(discrete_features)):
    sns.countplot(ax=ax[i, 0], x=discrete_features[i], data=data)
    sns.countplot(ax=ax[i, 1], x=discrete_features[i], hue='DEATH_EVENT', data=data)
fig.tight_layout(pad=1)
plt.show()
```



In [8]:
```
sns.countplot(x='DEATH_EVENT', data=data)
```

Out[8]:
```
<AxesSubplot:xlabel='DEATH_EVENT', ylabel='count'>
```

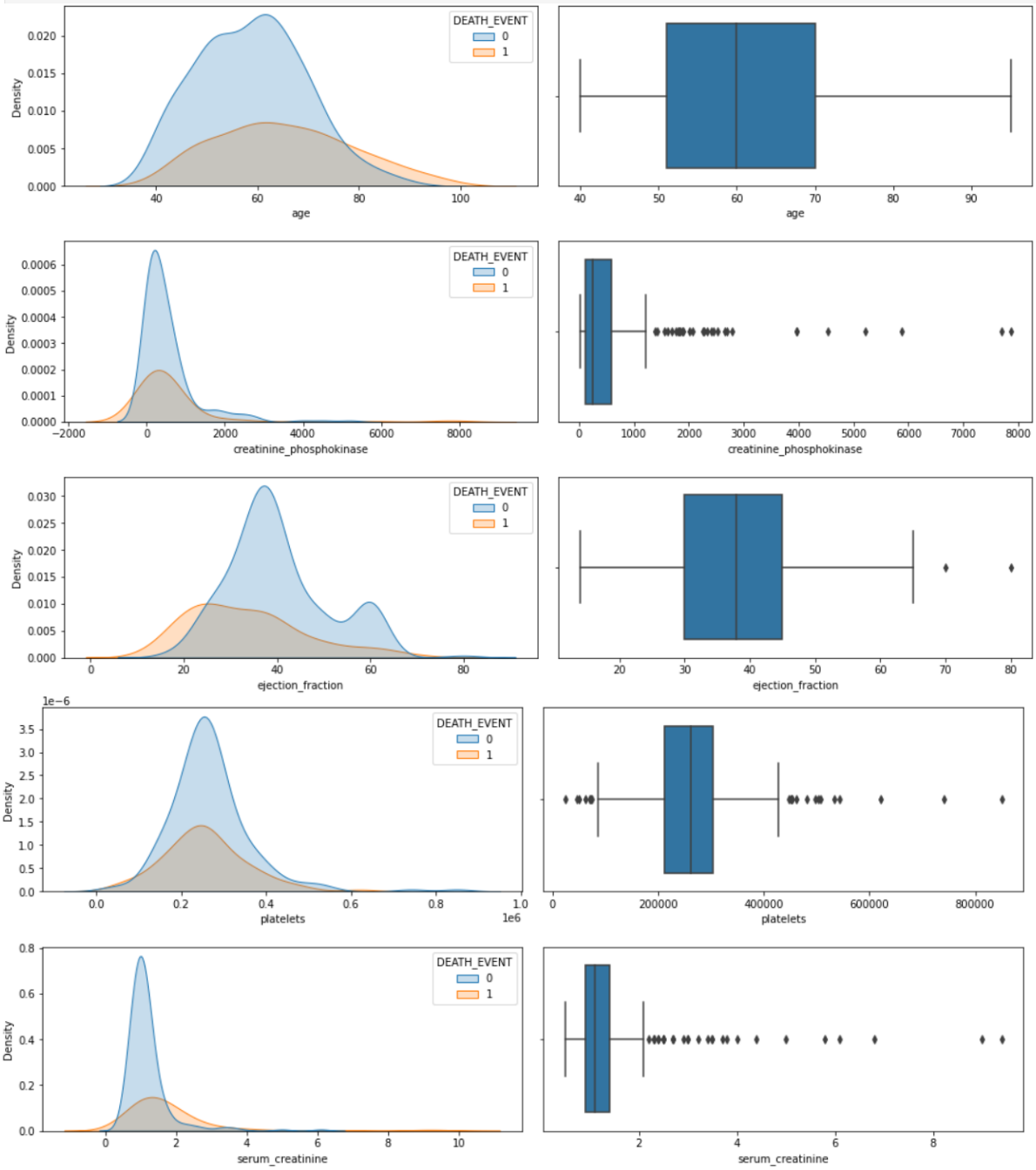


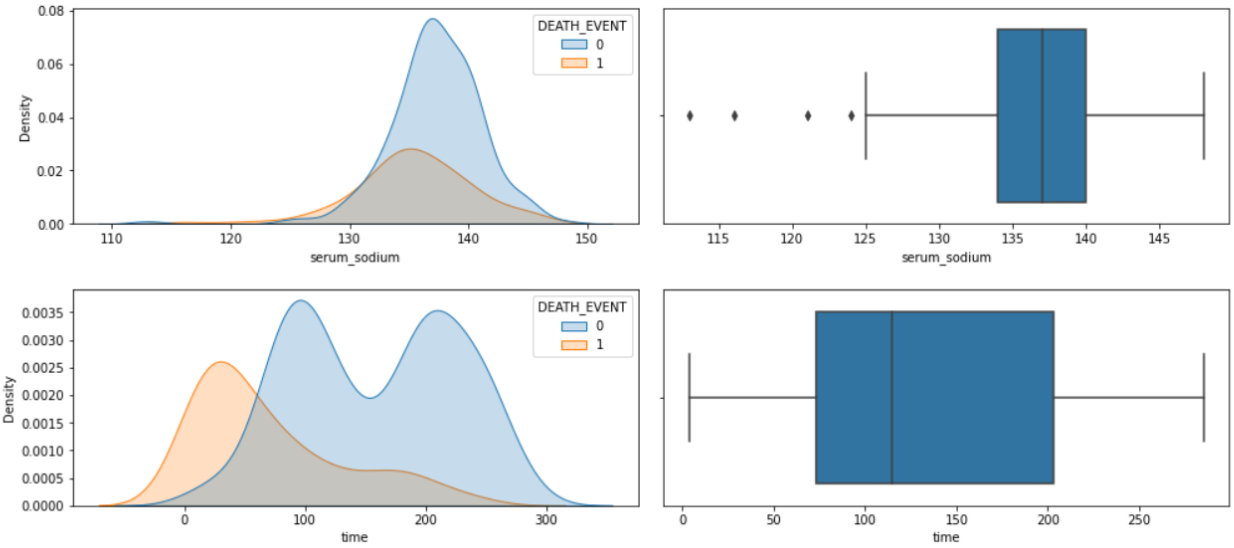
Observations

- There is an imbalance with the target variable, so we can apply cross validation technique with over sampling method compared to under sampling as the data size is small.

In [9]:
```
fig, ax = plt.subplots(len(continuous_features), 2, figsize=(14,22))

for i in range(len(continuous_features)):
    sns.kdeplot(ax=ax[i, 0], x=continuous_features[i], hue='DEATH_EVENT', data=data, fill = True)
    sns.boxplot(ax=ax[i, 1], x=continuous_features[i], data=data)
fig.tight_layout(pad=1)
plt.show()
```





Hypothesis Testing

Hypothesis 1 (H01): There is a relationship between diabetes and risk of heart failure as it is a major factor in real life.

Alternate Hypothesis (Ha1) There is no relation between diabetes and heart failure.

In [10]:
```
stats.ttest_ind(data['diabetes'], data['DEATH_EVENT'])
```

Out[10]:
```
Ttest_indResult(statistic=2.465229367999965, pvalue=0.013973299559289321)
```

Result: As p-value is less than 0.5, null hypothesis 1 is rejected.

Hypothesis 2 (H02): There is a relationship between smoking and risk of heart failure as smoking relates to bunch of diseases.

Alternate Hypothesis (Ha2) There is no relation between smoking and heart failure.

In [11]:
```
stats.ttest_ind(data['smoking'], data['DEATH_EVENT'])
```

Out[11]:
```
Ttest_indResult(statistic=0.0, pvalue=1.0)
```

Result: As p-value is greater than 0.5, null hypothesis 2 is accepted.

Hypothesis 3 (H03): There is a some connection between high BP and risk of heart failure as high BP stress on the heart functioning thus might affecting the patients predictions.

Alternate Hypothesis (Ha3) There is no relation between high BP and heart failure.

In [12]:
```
stats.ttest_ind(data['high_blood_pressure'], data['DEATH_EVENT'])
```

Out[12]:
```
Ttest_indResult(statistic=0.7782029809583625, pvalue=0.4367581827273743)
```

Result: As p-value is less than 0.5, null hypothesis 3 is rejected.

Future Scope

- Will try out different classifiers and selecting one with highest recall score.
- Recall represents the False Negative values which is very crucial in medical diagnosis.
- As the dataset is imbalanced, need to tackle the major issue.
- Alongwith that data rows are less, so will opt for oversampling or SMOTE or weighted class method.
- Feature selection is also crucial as some features are proven to be more prominent based on EDA, thus contributing more rather than selecting and processing all.

Notebook File available at: GitHub

Thank You for reviewing,
Aditya Mahimkar