Capstone Project-3

Exploratory Data Analysis (EDA) Project

Project-Heart Failure Analysis

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

1) ***Load the file***

```
hf=pd.read_csv("/content/heart_failure_clinical_records_dataset.csv")
```

2) ***Print first 5 rows of data***

```
hf.head()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine
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

Next steps: Generate code with hf | View recommended plots | New interactive sheet

3) ***Print last 5 rows of data***

```
hf.tail()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinin
294	62.0	0	61	1	38	1	155000.0	1.
295	55.0	0	1820	0	38	0	270000.0	1.
296	45.0	0	2060	1	60	0	742000.0	0.
297	45.0	0	2413	0	38	0	140000.0	1.
298	50.0	0	196	0	45	0	395000.0	1.

4) **Basic cleaning of data** (Checking null values,Missing values)

```
# Check for null values
hf.isnull().sum()
```

	0
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

```
# Drop rows with missing values
hf=hf.dropna()
```

5) ***Some information:***

1.Anemia -----> 0-No ; 1-Yes

2.diabetes-----> 0-No; 1-Yes

3.high_blood_preasure-----> 0-No; 1-Yes

4.Sex-----> 0-Female; 1-Male

5.Smoking-----> 0-No; 1-Yes

6.DEATH_EVENT------> 0-No; 1-Yes

6) ***Get some info on the data set:***

```
hf.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
```

7) ***Remove un-needed data-time column:***

Start coding or generate with AI.

8) ***Get some description of the data:***

```
hf.describe()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets
count	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000
mean	60.833893	0.431438	581.839465	0.418060	38.083612	0.351171	263358.029264
std	11.894809	0.496107	970.287881	0.494067	11.834841	0.478136	97804.236869
min	40.000000	0.000000	23.000000	0.000000	14.000000	0.000000	25100.000000
25%	51.000000	0.000000	116.500000	0.000000	30.000000	0.000000	212500.000000
50%	60.000000	0.000000	250.000000	0.000000	38.000000	0.000000	262000.000000
75%	70.000000	1.000000	582.000000	1.000000	45.000000	1.000000	303500.000000
max	95.000000	1.000000	7861.000000	1.000000	80.000000	1.000000	850000.000000

9) ***Shape of the dataset:***

```
(hf.shape)
```

```
(299, 13)
```

10) ***Find how many records are there (value_counts)***

1.gender

2.high blood pressure

3.diabetes

4.smoking

5.death_event

```
#1-gender
hf['sex'].value_counts()
```

sex	count
1	194
0	105

dtype: int64

```
#2-high blood pressure
hf['high_blood_pressure'].value_counts()
```

high_blood_pressure	count
0	194
1	105

dtype: int64

```
#3-diabetes
hf['diabetes'].value_counts()
```

diabetes	count
0	174
1	125

dtype: int64

```
#4-smoking
hf['smoking'].value_counts()
```

smoking	count
0	203
1	96

dtype: int64

```
#5-Death Event
hf['DEATH_EVENT'].value_counts()
```

DEATH_EVENT	count
0	203
1	96

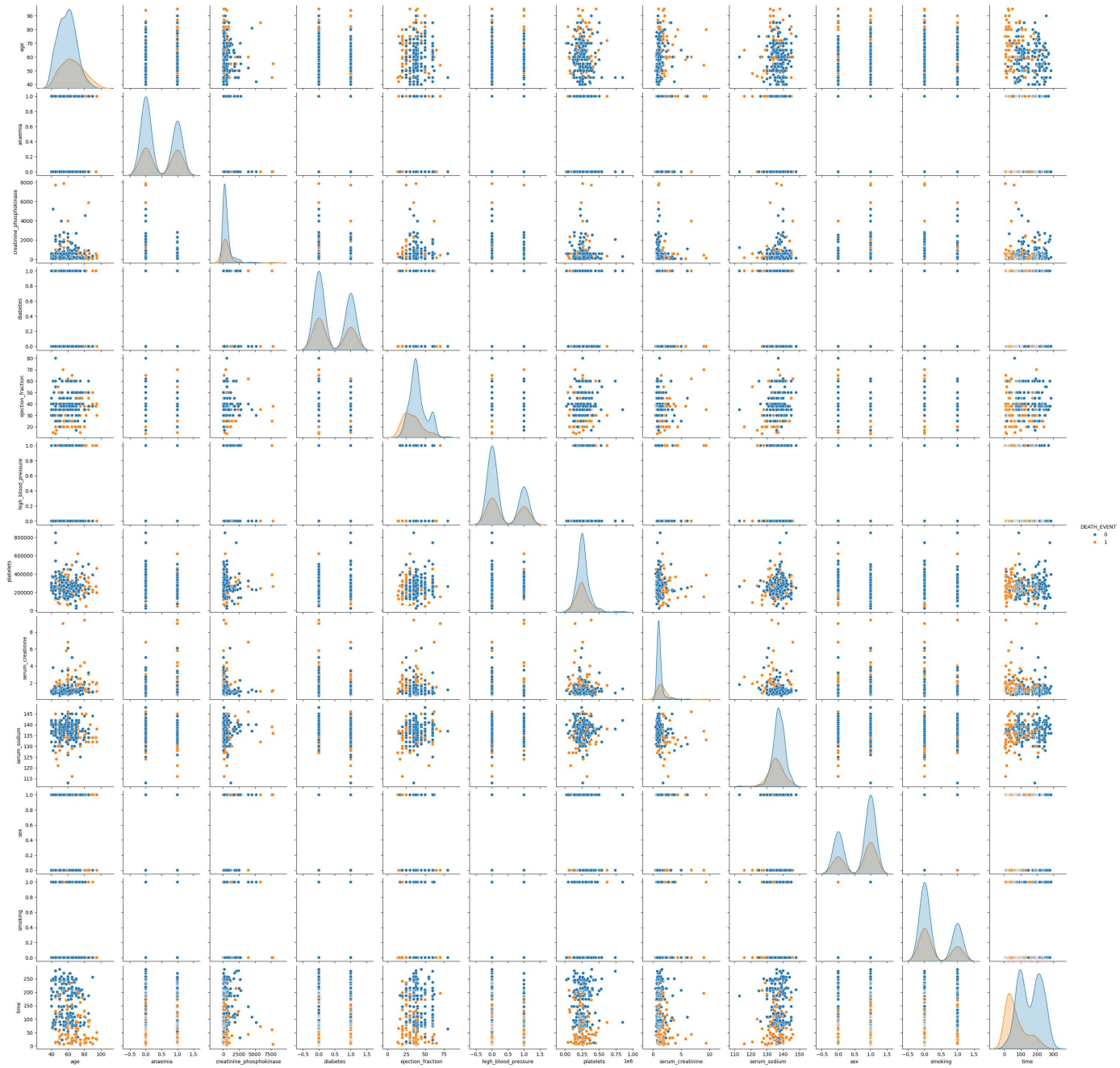
dtype: int64

Visualization:

```
import seaborn as sns
import matplotlib.pyplot as plt
```

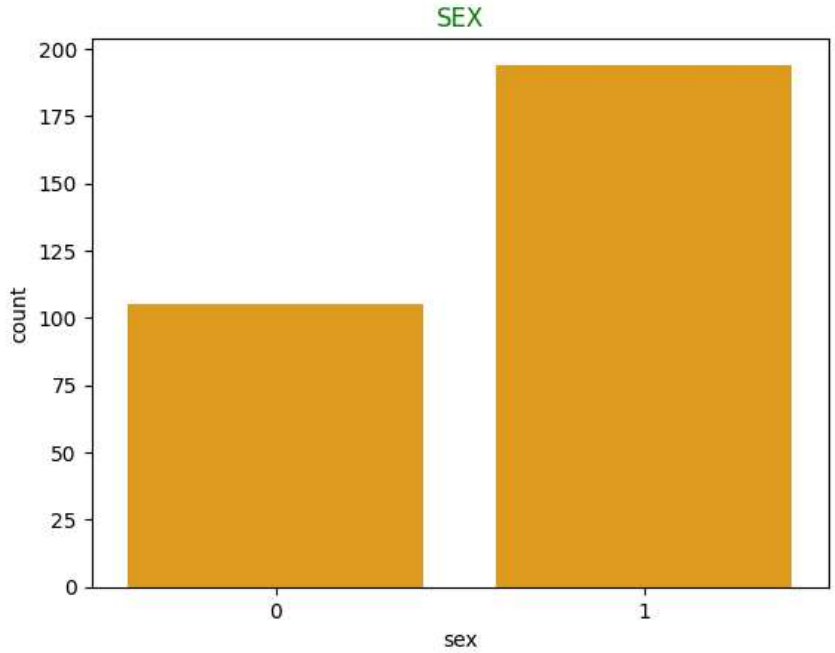
1) ***Pairplot with relation to death event:***

```
sns.pairplot(hf,hue='DEATH_EVENT')
plt.title('DEATH EVENT',color='pink')
plt.show()
```



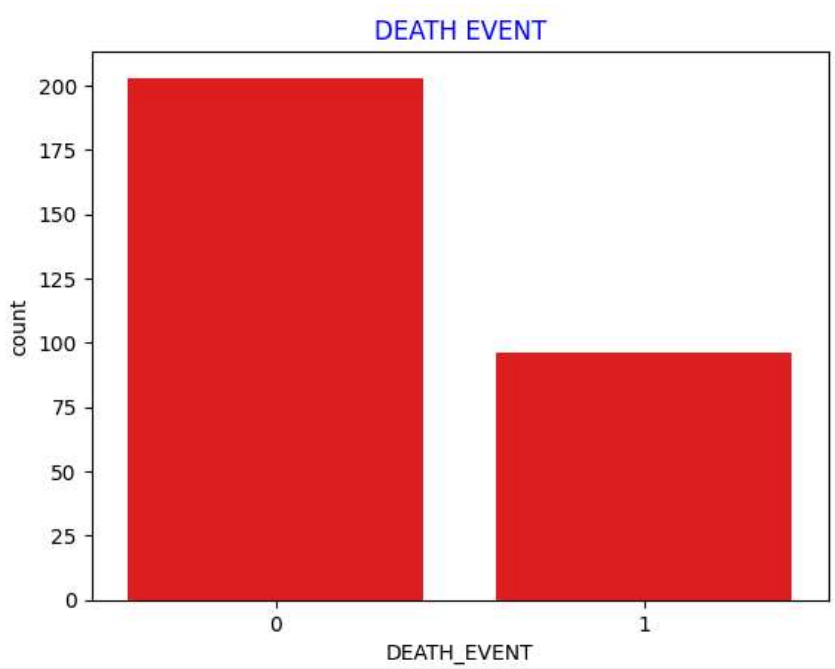
2) ***Bar plot for 'sex':***

```
sns.countplot(x='sex',data=hf,color='orange')
plt.title('SEX',color='green')
plt.show()
```



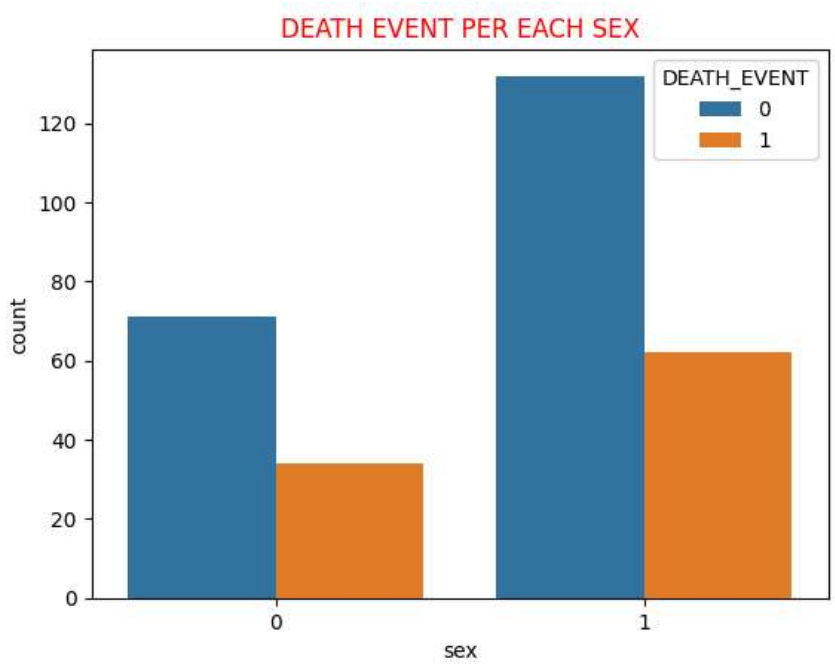
3) **Bar plot for 'DEATH_EVENT':**

```
sns.countplot(x='DEATH_EVENT',data=hf,color='red')
plt.title('DEATH EVENT',color='blue')
plt.show()
```



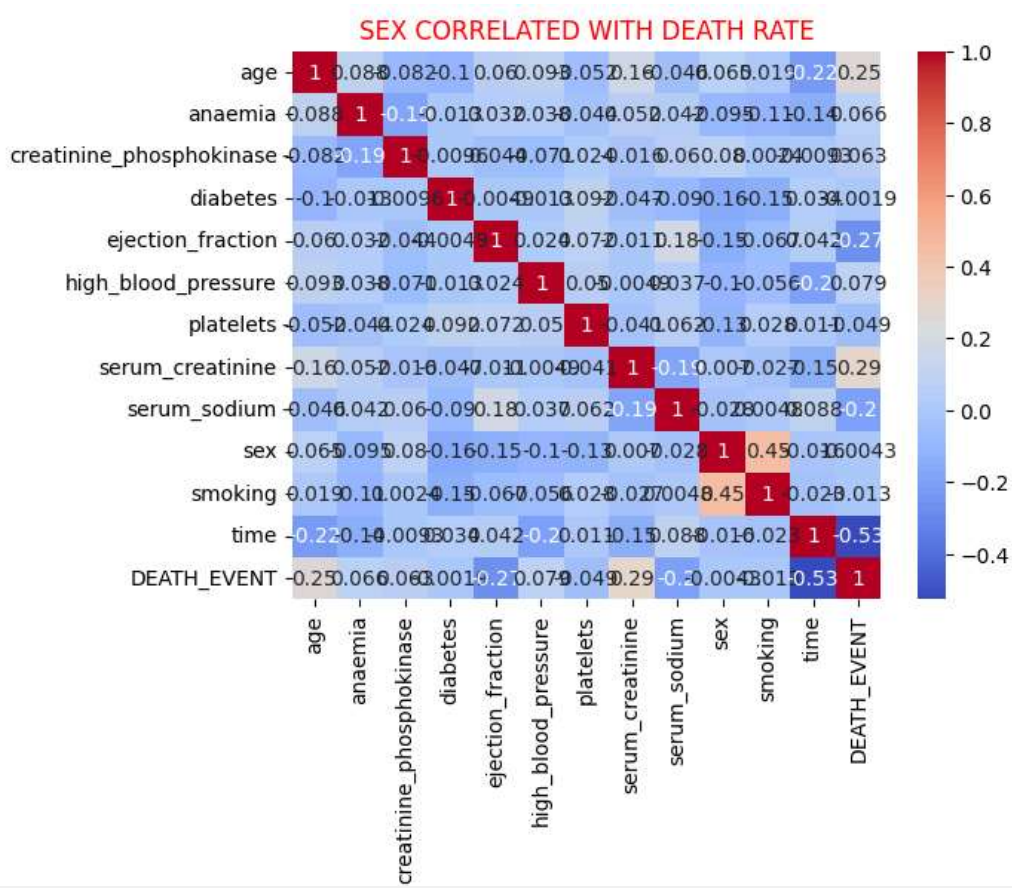
4) ***Death event per each sex using barplot:***

```
sns.countplot(x='sex',hue='DEATH_EVENT',data=hf)
plt.title('DEATH EVENT PER EACH SEX',color='red')
plt.show()
```



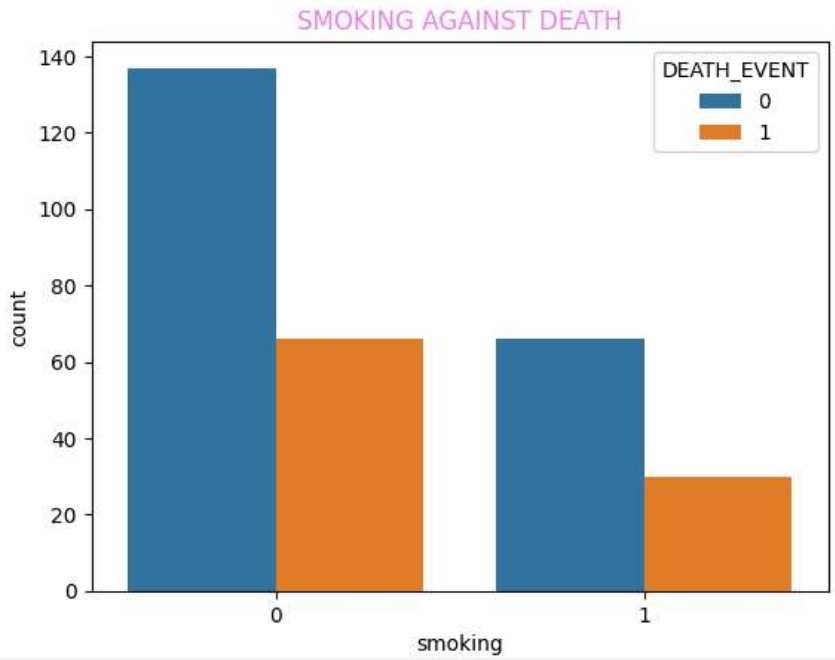
5) ***Sex correlated with Death rate using heatmap:***

```
corr=hf.corr()
sns.heatmap(corr,annot=True,cmap='coolwarm')
plt.title('SEX CORRELATED WITH DEATH RATE',color='red')
plt.show()
```



6) ***Smoking against Death using barplot:***

```
sns.countplot(x='smoking',hue='DEATH_EVENT',data=hf)
plt.title('SMOKING AGAINST DEATH',color='violet')
plt.show()
```



7)***High blood pressure with age using catplot:***

```
sns.catplot(x='high_blood_pressure',y='age',data=hf)
plt.title('HIGH BLOOD PRESSURE WITH AGE',color='green')
plt.show()
```