DATA ANALYSING

1. Load the file

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
hfa_df = pd.read_csv('/content/heart_failure_clinical_records_dataset.csv')
```

2. Print first 5 rows of data

```
hfa_df.head()
```

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

3. Print last 5 rows of data

```
hfa_df.tail()
```

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

4. You have to do the basic cleaning of data for checking null values, missing values etc.

```
hfa_df.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

6. Get some info on the dataset

```
hfa_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
```

7. Remove un-needed data - time colum

```
hfa_df = hfa_df.drop('time', axis=1)
```

8. Get some description of the data.

```
hfa_df.describe()
```

	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatini
count	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000000	299.000
mean	60.833893	0.431438	581.839465	0.418060	38.083612	0.351171	263358.029264	1.393
std	11.894809	0.496107	970.287881	0.494067	11.834841	0.478136	97804.236869	1.034
min	40.000000	0.000000	23.000000	0.000000	14.000000	0.000000	25100.000000	0.500
25%	51.000000	0.000000	116.500000	0.000000	30.000000	0.000000	212500.000000	0.900
50%	60.000000	0.000000	250.000000	0.000000	38.000000	0.000000	262000.000000	1.100
75%	70.000000	1.000000	582.000000	1.000000	45.000000	1.000000	303500.000000	1.400
max	95.000000	1.000000	7861.000000	1.000000	80.000000	1.000000	850000.000000	9.400

9. Shape of the Dataset

```
hfa_df.shape
```

```
(299, 12)
```

10. Find how many gender, high blood pressure, diabetes,smoking, death_event records are there. (value_counts)

```
print(hfa_df['sex'].value_counts())
print(hfa_df['high_blood_pressure'].value_counts())
print(hfa_df['diabetes'].value_counts())
print(hfa_df['smoking'].value_counts())
print(hfa_df['DEATH_EVENT'].value_counts())
```

```
sex
1    194
0    105
Name: count, dtype: int64
high_blood_pressure
0    194
1    105
Name: count, dtype: int64
diabetes
0    174
1    125
Name: count, dtype: int64
smoking
0    203
1     96
Name: count, dtype: int64
DEATH_EVENT
0    203
1     96
Name: count, dtype: int64
```

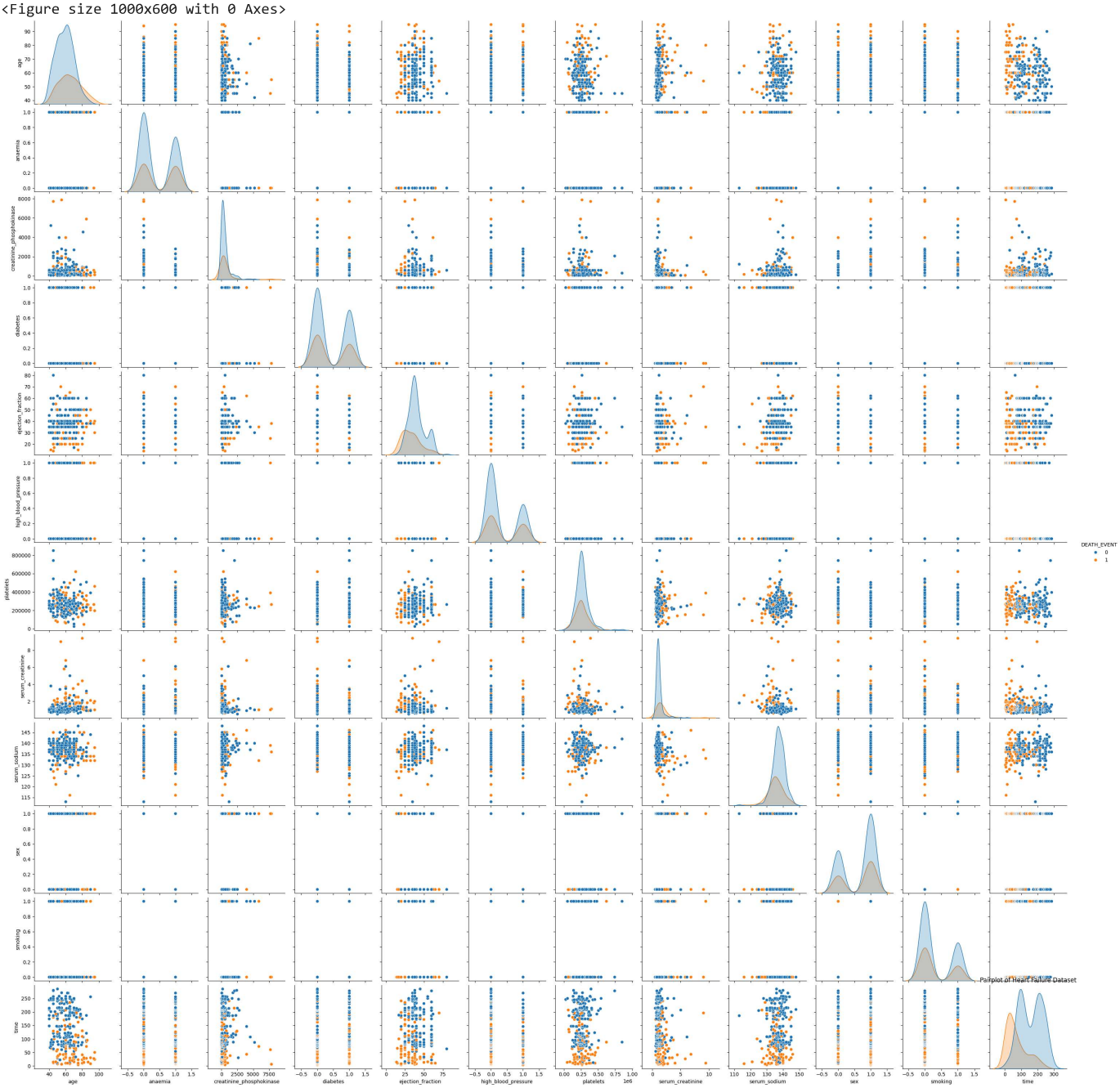
DATA VISUALIZATION

---

1. Show the relationship of the whole dataset (with relation to death event) using pairplot.

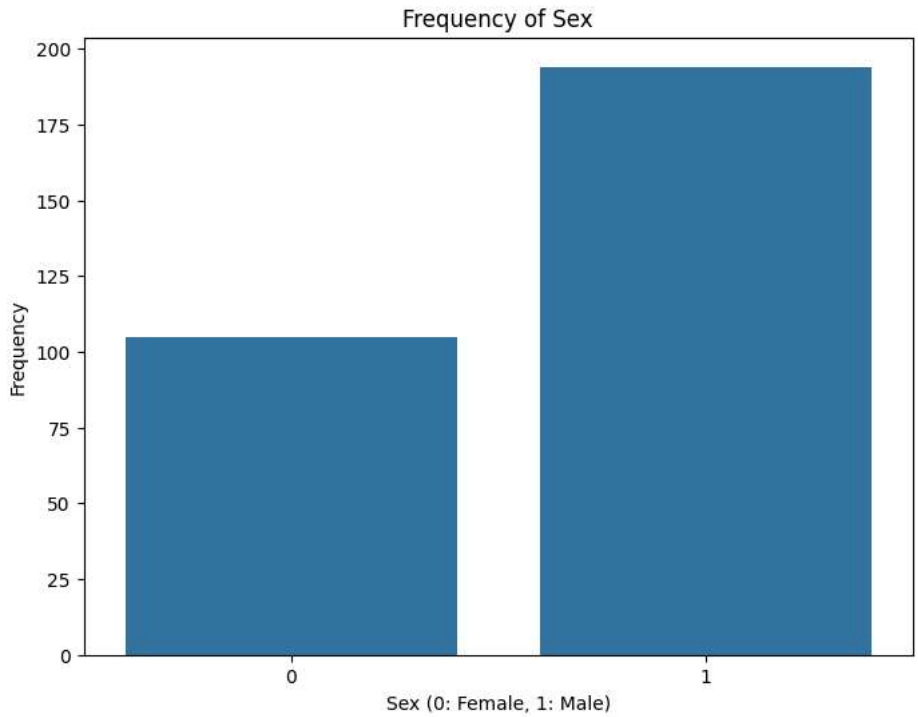
```
plt.figure(figsize=(10, 6))
sns.pairplot(hfa_df, hue='DEATH_EVENT')
plt.title('Pairplot of Heart Failure Dataset')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```



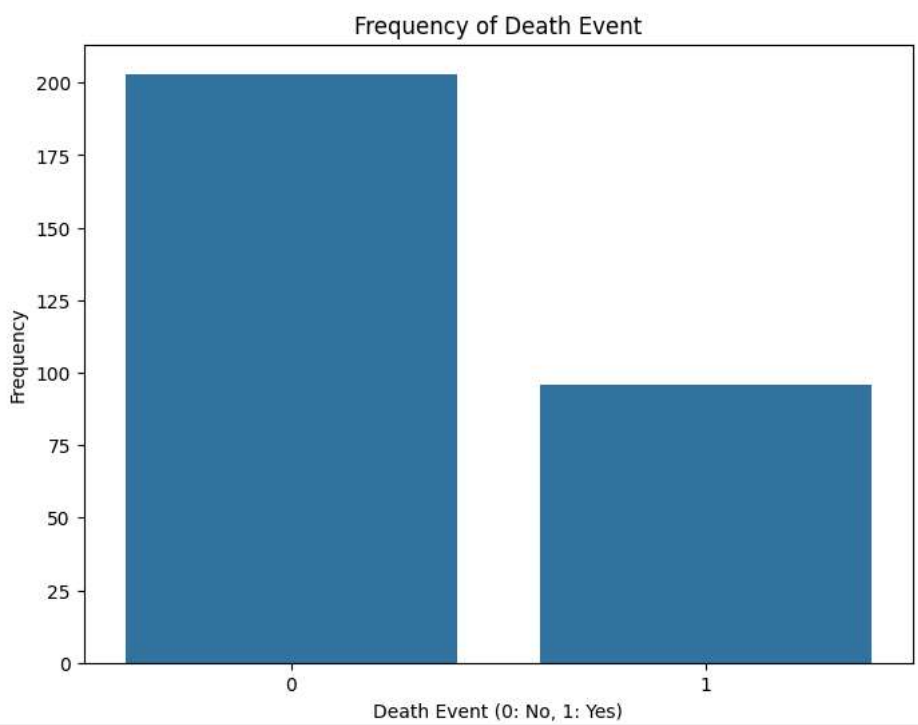
2. Showing the relationship between categoric variable "sex" and its frequency using bar plot.

```
plt.figure(figsize=(8, 6))
sns.countplot(x='sex', data=hfa_df)
plt.title('Frequency of Sex')
plt.xlabel('Sex (0: Female, 1: Male)')
plt.ylabel('Frequency')
plt.show()
```



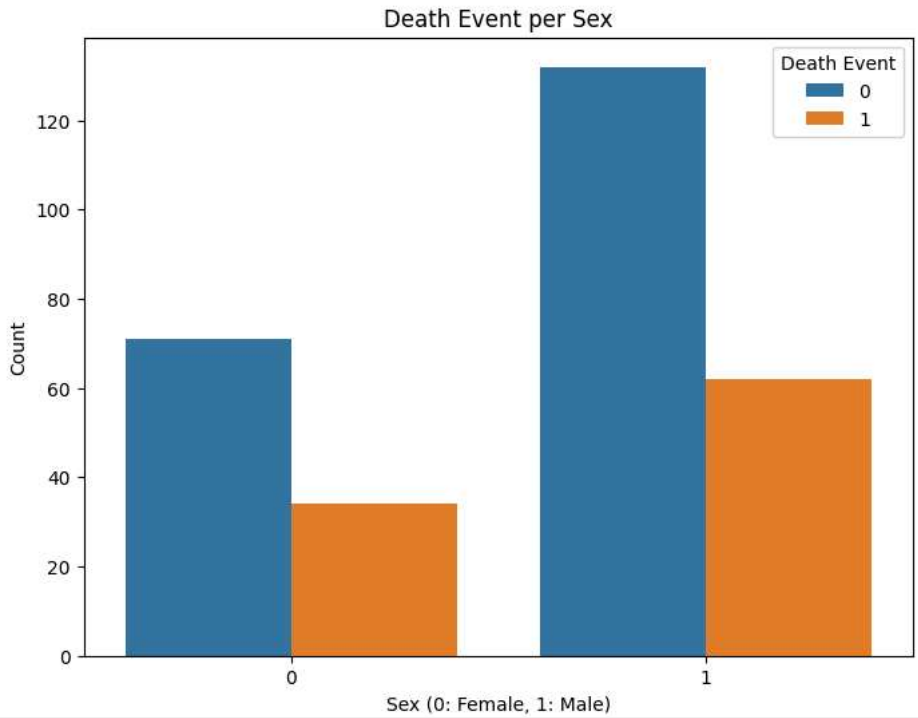
3. Showing the relationship between categoric variable "death_event" and its frequency using bar plot.

```
plt.figure(figsize=(8, 6))
sns.countplot(x='DEATH_EVENT', data=hfa_df)
plt.title('Frequency of Death Event')
plt.xlabel('Death Event (0: No, 1: Yes)')
plt.ylabel('Frequency')
plt.show()
```



4. Death event per each sex using bar plot

```
plt.figure(figsize=(8, 6))
sns.countplot(x='sex', hue='DEATH_EVENT', data=hfa_df)
plt.title('Death Event per Sex')
plt.xlabel('Sex (0: Female, 1: Male)')
plt.ylabel('Count')
plt.legend(title='Death Event', loc='upper right')
plt.show()
```

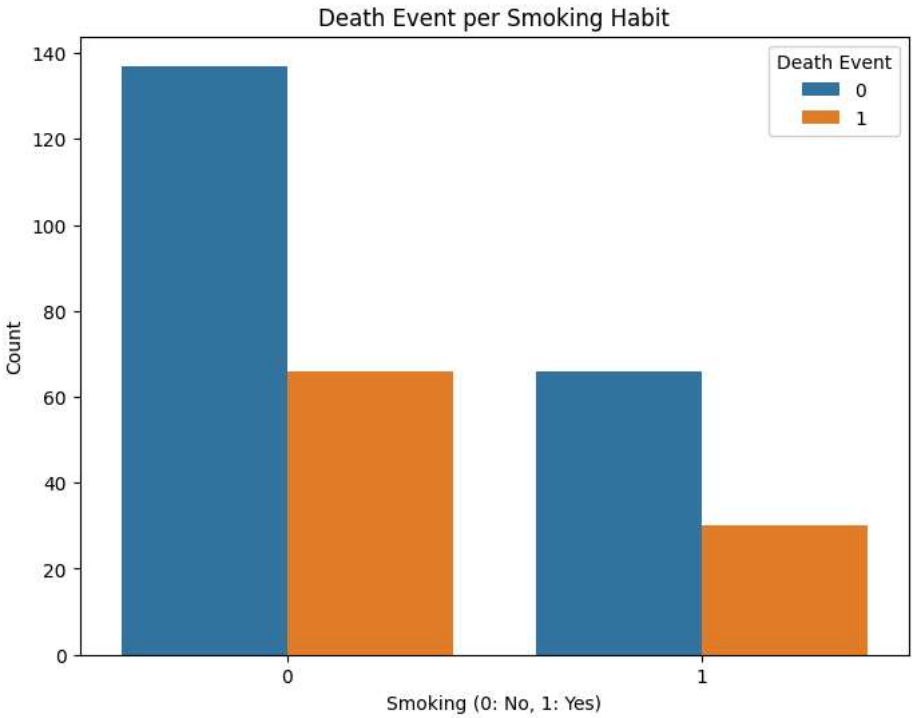


5. Sex correlated with Death rate (use heatmap)

```
correlation_matrix = hfa_df[['sex', 'DEATH_EVENT']].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation between Sex and Death Rate')
plt.show()
```

6. Smoking against Death using bar plot

```
plt.figure(figsize=(8, 6))
sns.countplot(x='smoking', hue='DEATH_EVENT', data=hfa_df)
plt.title('Death Event per Smoking Habit')
plt.xlabel('Smoking (0: No, 1: Yes)')
plt.ylabel('Count')
plt.legend(title='Death Event', loc='upper right')
plt.show()
```



7. High blood pressure with age using catplot.

```
plt.figure(figsize=(10, 6))
sns.catplot(x='age', y='high_blood_pressure', data=hfa_df, kind='box')
plt.title('High Blood Pressure with Age')
plt.xlabel('Age')
plt.ylabel('High Blood Pressure (0: No, 1: Yes)')
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```