DMBI - Experiment 3

Mansi Davkhar
D15C / 52

Aim: To perform Exploratory Data Analysis and visualization using python.

Theory:
Exploratory Data Analysis (EDA) is the process of exploring, summarizing, and visualizing data to understand its main characteristics before applying statistical models or machine learning. It helps researchers **detect underlying structures, spot anomalies, identify relationships, and test hypotheses**.

The major steps include:

1. **Descriptive Statistics**
   - Provides numerical summaries such as mean, median, variance, min/max values, and standard deviation.
   - Helps detect skewness, outliers, and unusual distributions.

2. **Target Variable Analysis**
   - The dependent variable (here: `heart_disease`) is visualized with count plots to check class balance.
   - If the dataset is highly imbalanced, it may affect classification performance.

3. **Correlation Analysis**
   - Pearson's correlation coefficient is calculated between numeric features.
   - A **heatmap** helps identify strong positive/negative correlations (e.g., thalach vs. age, chol vs. bmi).
   - Useful for detecting multicollinearity or redundant features.

4. **Feature Distribution Analysis**
   - Histograms/KDE plots show how features like `age`, `cholesterol`, and `bmi` are distributed (normal, skewed, multimodal).
   - Boxplots grouped by target show how continuous features vary between patients with and without heart disease.

5. **Categorical Feature Analysis**
    - Countplots and bar charts show how categorical features (e.g., `sex`, `cp`, `thal`) are distributed across target classes.
    - This helps assess the predictive power of categorical variables (e.g., chest pain type has strong association with heart disease).

6. **Multivariate Visualization**
    - Pairplots allow simultaneous visualization of multiple variables, highlighting clusters and class separation.
    - Useful to see which combinations of features separate patients with vs. without heart disease.

Importance:

- Provides deeper insight into dataset structure.
- Helps select features that are most relevant for prediction.
- Reveals outliers or errors that might need special treatment.
- Builds intuition about how independent variables influence the target outcome.

Conclusion:

- Some features (like chest pain type, thalach, and oldpeak) showed strong separation between heart disease and non–heart disease patients.
- Categorical features like `sex`, `smoking`, and `diabetes` showed imbalances but are still relevant for prediction.
- Correlation heatmap revealed moderate correlations between variables (e.g., age vs. thalach).
- No extreme imbalance in the target variable was observed, which supports fair classification modeling.
- Overall, EDA confirmed that the dataset contains meaningful signals for predicting heart disease.

Output:

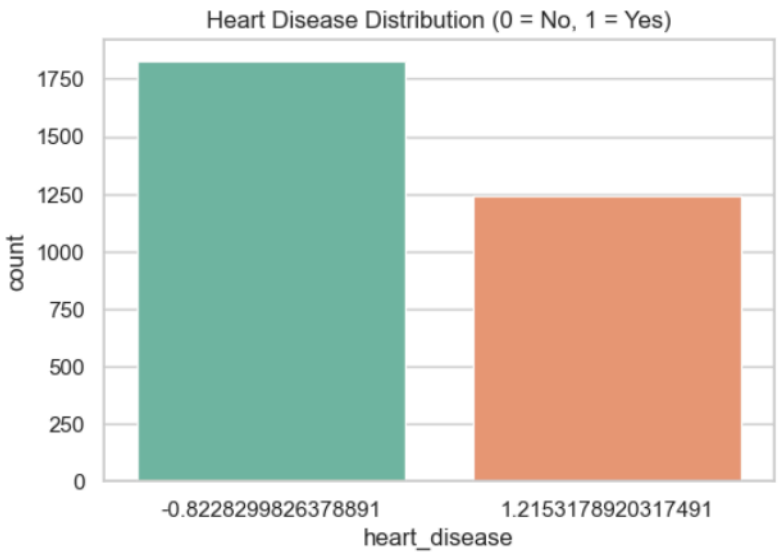
[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
df = pd.read_csv("heart_disease_cleaned.csv")

# General settings
pd.set_option("display.max_columns", None)
sns.set(style="whitegrid", palette="muted")
```

[3]:
```
# Summary stats for numeric columns
df.describe().T
```

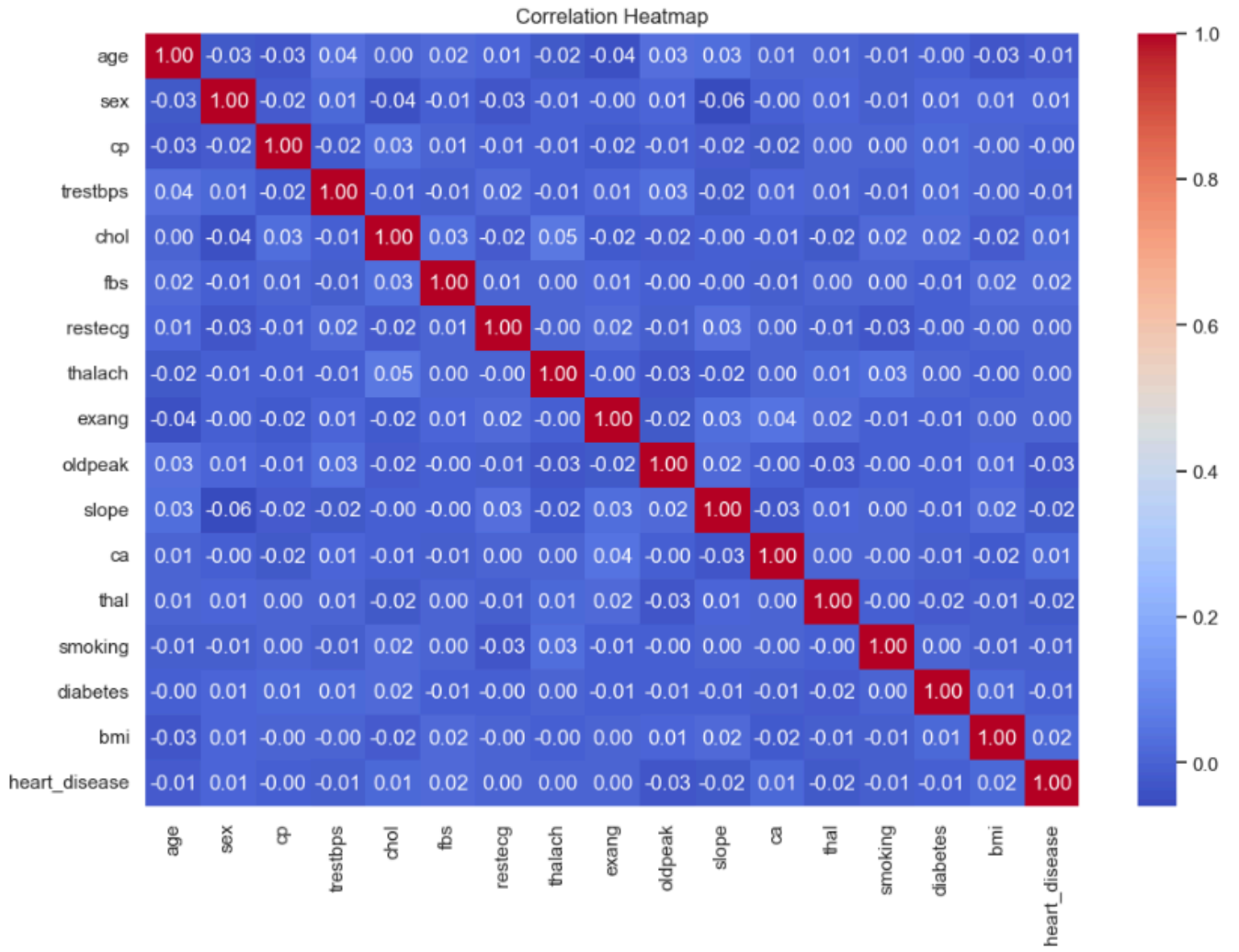
[3]:

	count	mean	std	min	25%	50%	75%	max
age	3069.0	2.604629e-16	1.000163	-1.714750	-0.839175	0.036399	0.839009	1.714584
sex	3069.0	-9.723948e-17	1.000163	-1.118034	-1.118034	0.894427	0.894427	0.894427
cp	3069.0	2.778271e-17	1.000163	-1.341819	-0.441110	-0.441110	0.459599	1.360308
trestbps	3069.0	4.323684e-16	1.000163	-1.750356	-0.864200	0.021955	0.844813	1.730969
chol	3069.0	-1.834816e-16	1.000163	-1.683722	-0.900501	0.018934	0.870262	1.721590
fbs	3069.0	2.749330e-17	1.000163	-0.412893	-0.412893	-0.412893	-0.412893	2.421936
restecg	3069.0	-2.164736e-16	1.000163	-1.236922	-1.236922	-0.005217	1.226487	1.226487
thalach	3069.0	2.778271e-17	1.000163	-1.733889	-0.857225	-0.003631	0.849963	1.726627
exang	3069.0	8.219051e-17	1.000163	-0.462605	-0.462605	-0.462605	-0.462605	2.161673
oldpeak	3069.0	-2.060551e-16	1.000163	-1.769218	-0.881572	0.006074	0.838242	1.670410

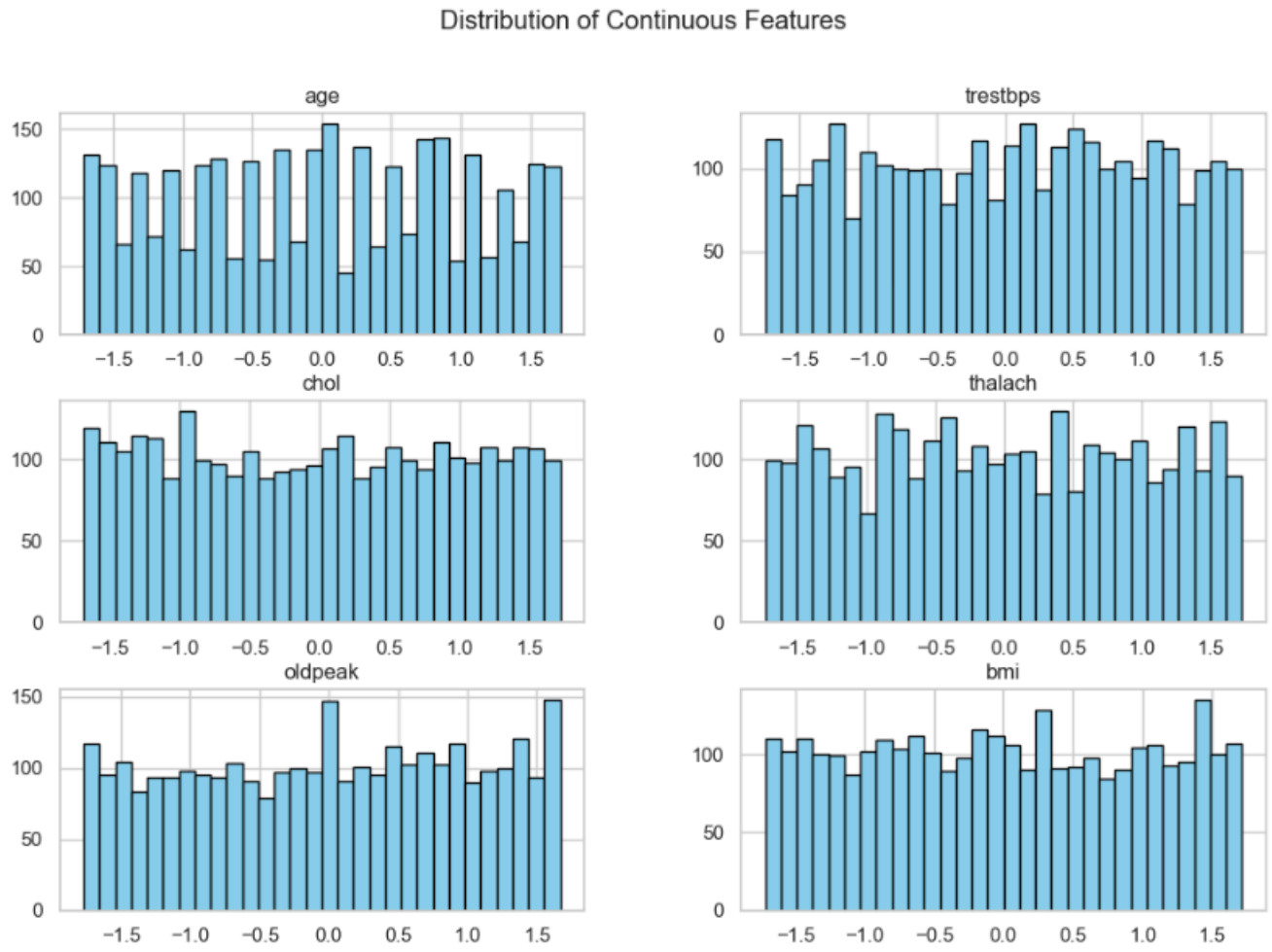
[5]:
```
plt.figure(figsize=(6,4))
sns.countplot(x="heart_disease", hue="heart_disease", data=df, palette="Set2", legend=False)
plt.title("Heart Disease Distribution (0 = No, 1 = Yes)")
plt.show()
```



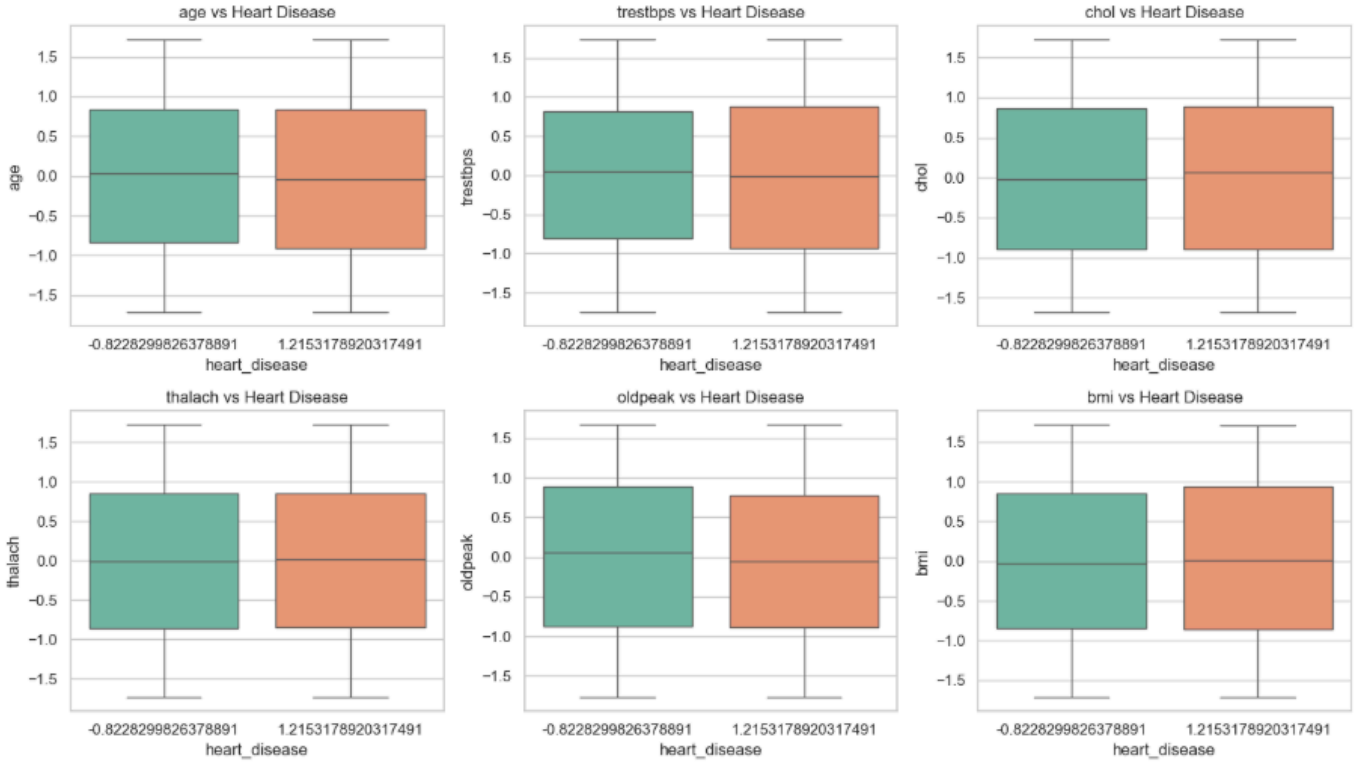
[6]:
```
plt.figure(figsize=(12,8))
corr = df.corr()
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```



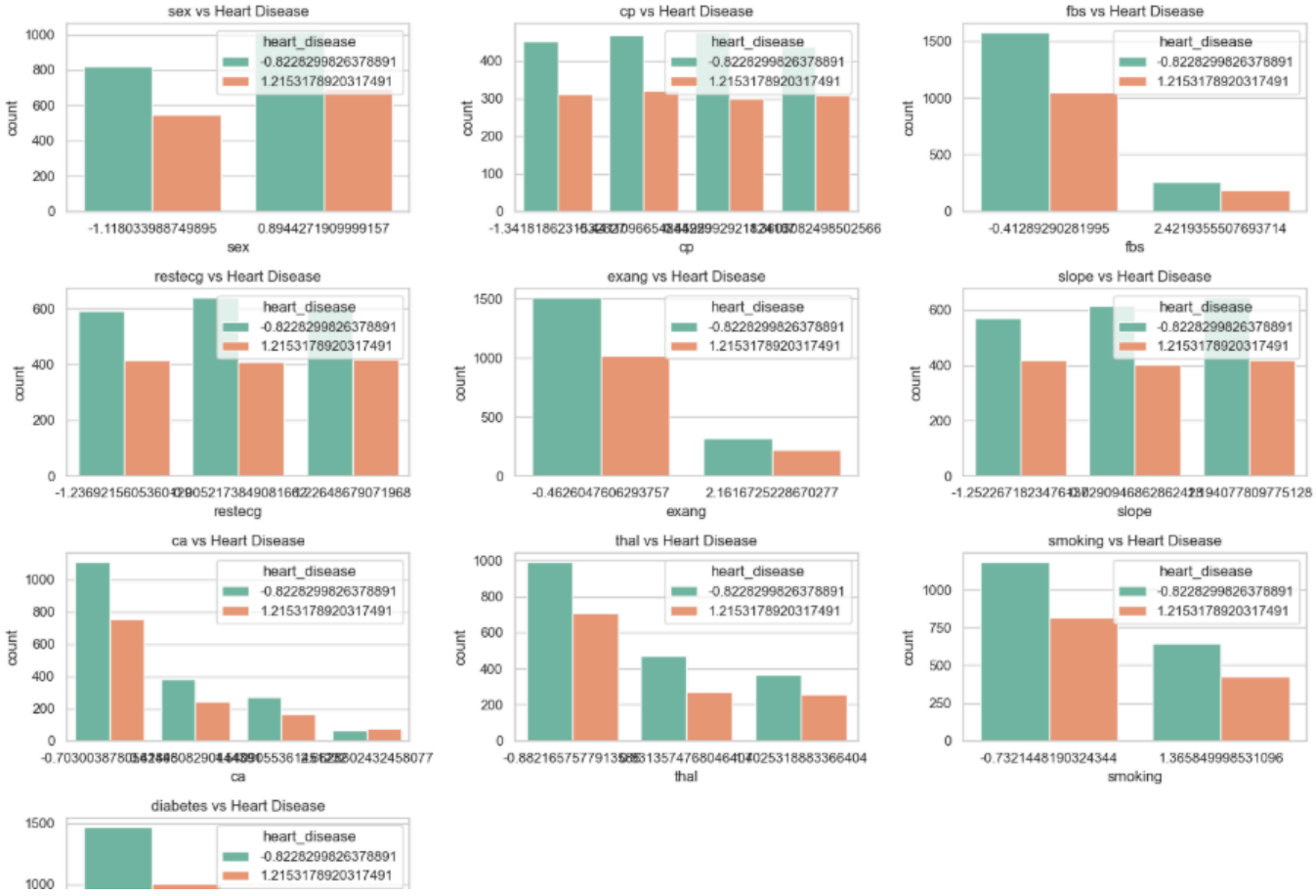
[7]:
```
numeric_cols = ["age", "trestbps", "chol", "thalach", "oldpeak", "bmi"]

df[numeric_cols].hist(bins=30, figsize=(12,8), color="skyblue", edgecolor="black")
plt.suptitle("Distribution of Continuous Features")
plt.show()
```



[8]:
```
plt.figure(figsize=(14,8))
for i, col in enumerate(numeric_cols, 1):
    plt.subplot(2,3,i)
    sns.boxplot(x="heart_disease", y=col, data=df, hue="heart_disease", palette="Set2", legend=False)
    plt.title(f"{col} vs Heart Disease")
plt.tight_layout()
plt.show()
```



[9]:
```
categorical_cols = ["sex", "cp", "fbs", "restecg", "exang", "slope",
                    "ca", "thal", "smoking", "diabetes"]

plt.figure(figsize=(15,12))
for i, col in enumerate(categorical_cols, 1):
    plt.subplot(4,3,i)
    sns.countplot(x=col, hue="heart_disease", data=df, palette="Set2")
    plt.title(f"{col} vs Heart Disease")
plt.tight_layout()
plt.show()
```



[11]:
```
sns.pairplot(df[["age", "trestbps", "chol", "thalach", "bmi", "heart_disease"]],
             hue="heart_disease", palette="Set2", diag_kind="kde")
plt.show()
```