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

```
# read the csv file
data = pd.read_csv('/content/Heart Disease data.csv')
```

```
data.head()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

```
## DATA EXPLORATION
```

```
data.target.value_counts()
```

```
1    526
0    499
Name: target, dtype: int64
```

```
# checking for null values
data.isna().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

```
# checking for duplicate values
data[data.duplicated()]
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	c
15	34	0	1	118	210	0	1	192	0	0.7	2	
31	50	0	1	120	244	0	1	162	0	1.1	2	
43	46	1	0	120	249	0	0	144	0	0.8	2	
55	55	1	0	140	217	0	1	111	1	5.6	0	
61	66	0	2	146	278	0	0	152	0	0.0	1	
...	...	...	...	...	...	...	...	...	...	...	...	.
1020	59	1	1	140	221	0	1	164	1	0.0	2	
1021	60	1	0	125	258	0	0	141	1	2.8	1	
1022	47	1	0	110	275	0	0	118	1	1.0	1	
1023	50	0	0	110	254	0	0	159	0	0.0	2	
1024	54	1	0	120	188	0	1	113	0	1.4	1	

723 rows × 14 columns

```
data.drop_duplicates(inplace=True)
```

```
data.nunique()
```

```
age          41
sex           2
cp            4
trestbps     49
chol        152
fbs           2
restecg       3
thalach      91
exang         2
oldpeak      40
slope         3
ca            5
thal          4
target        2
dtype: int64
```

```
data.cp.value_counts()
```

```
0    143
2     86
1     50
3     23
Name: cp, dtype: int64
```

```
data.ca.value_counts()
```

```
0    175
1     65
2     38
3     20
4      4
Name: ca, dtype: int64
```

```
# those values are marked as NAN , so that it can be removed during removal of missing values phase
```

```
data.loc[data['ca']==4, 'ca']=np.NaN    # marking those values as NAN values
```

```
data['ca'].unique()
```

```
array([ 2.,  0.,  1.,  3., nan])
```

```
data.thal.value_counts()
```

```
2    165
3    117
```

```
1     18
0      2
Name: thal, dtype: int64
```

```
# as per data dict, feature -thal has 3 unique values, but dataset has two 0 values , which needs to be removed
```

```
data.loc[data['thal']==0]
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca
14	52	1	0	128	204	1	1	156	1	1.0	1	0.0
319	53	0	2	128	216	0	0	115	0	0.0	2	0.0

```
data.loc[data['thal']==0, 'thal']=np.NaN
```

```
data.thal.unique()
```

```
array([ 3.,  2.,  1., nan])
```

```
data.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          4
thal        2
target      0
dtype: int64
```

```
sns.distplot(data.target)
```

<ipython-input-78-b9023a40a00b>:1: UserWarning:

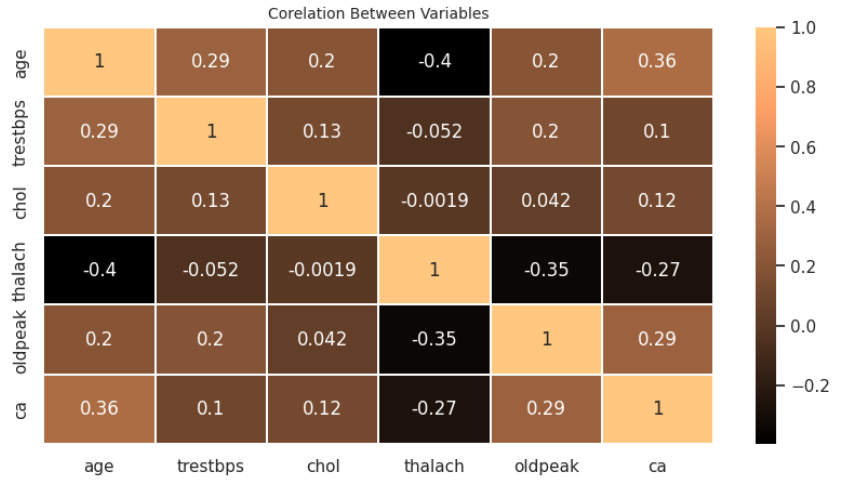
```
## Data Exploration
```

```
data.describe()
```

	age	sex	cp	trestbps	chol	fbs	reste
count	296.000000	296.000000	296.000000	296.00000	296.000000	296.00000	296.0000
mean	54.523649	0.679054	0.959459	131.60473	247.155405	0.14527	0.5236
std	9.059471	0.467631	1.034184	17.72662	51.977011	0.35297	0.5266
min	29.000000	0.000000	0.000000	94.00000	126.000000	0.00000	0.0000
25%	48.000000	0.000000	0.000000	120.00000	211.000000	0.00000	0.0000
50%	56.000000	1.000000	1.000000	130.00000	242.500000	0.00000	1.0000
75%	61.000000	1.000000	2.000000	140.00000	275.250000	0.00000	1.0000
max	77.000000	1.000000	3.000000	200.00000	564.000000	1.00000	2.0000

```
sns.set(style="white")
plt.rcParams['figure.figsize'] = (10, 5)
corrmat=sns.heatmap(data.corr(), annot = True, linewidths=.1, cmap="copper")
plt.title('Corelation Between Variables', fontsize = 10)
plt.show()
```

```
<ipython-input-149-5fd3d9ea4ba7>:3: FutureWarning: The default value of numeric_on
  corrmat=sns.heatmap(data.corr(), annot = True, linewidths=.1, cmap="copper")
```




```
# Renaming for better analysis and feature description
```

```
data['target'] = data.target.replace({1: "Possible CVD", 0: "No CVD"})
```

```
data['sex'] = data.sex.replace({1: "Male", 0: "Female"})
```

```
data['cp'] = data.cp.replace({0: "typical_angina",
                              1: "atypical_angina",
                              2:"non-anginal pain",
                              3: "asymptomatic"})
```

```
data['exang'] = data.exang.replace({1: "Yes", 0: "No"})
```

```
data['fbs'] = data.fbs.replace({1: "True", 0: "False"})
```

```
data['slope'] = data.slope.replace({0: "upslope", 1: "flat",2:"downslope"})
```

```
data['thal'] = data.thal.replace({1: "fixed_defect", 2: "reversable_defect", 3:"normal"})
```

```
data['restecg'] = data.restecg.replace({1: "Normal", 0: "Abnormal"})
```

```
data.head(1)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak
0	52	Male	typical angina	125	212	False	Normal	168	No	1.0

```
continuousFeatures = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

def outlier_treatment(data, drop=False):
    for eachFeature in continuousFeatures:
        featureData = data[eachFeature]
        Q1, Q3 = np.percentile(featureData, [25, 75])
        IQR = Q3 - Q1
        outlierCalc = 1.5 * IQR
        outliers = featureData[~((featureData >= Q1 - outlierCalc) & (featureData <= Q3 + outlierCalc))].index.tolist()
        if not drop:
            print('For the feature {}, No of Outliers is {}'.format(eachFeature, len(outliers)))
        if drop:
            data.drop(outliers, inplace=True)
            print('Outliers from {} feature removed'.format(eachFeature))

outlier_treatment(data[continuousFeatures], drop=True)
```

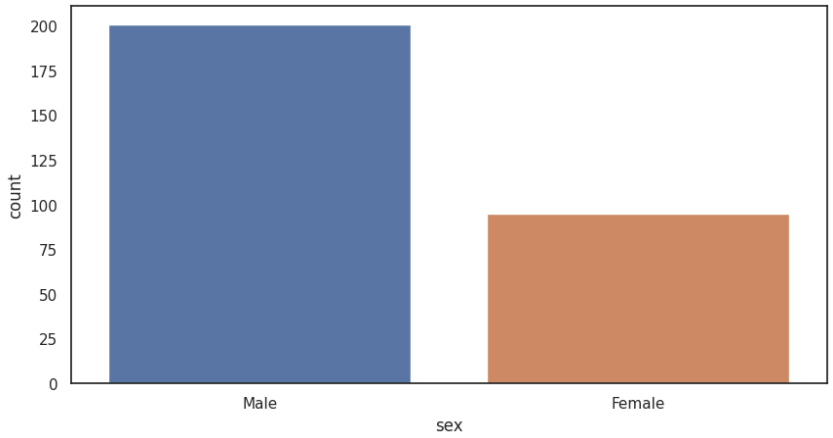
```
Outliers from age feature removed
Outliers from trestbps feature removed
Outliers from chol feature removed
Outliers from thalach feature removed
Outliers from oldpeak feature removed
<ipython-input-150-bd5de829edf8>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data.drop(outliers, inplace=True)
<ipython-input-150-bd5de829edf8>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data.drop(outliers, inplace=True)
<ipython-input-150-bd5de829edf8>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data.drop(outliers, inplace=True)
<ipython-input-150-bd5de829edf8>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data.drop(outliers, inplace=True)
<ipython-input-150-bd5de829edf8>:13: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data.drop(outliers, inplace=True)
```

```
data.skew()
```

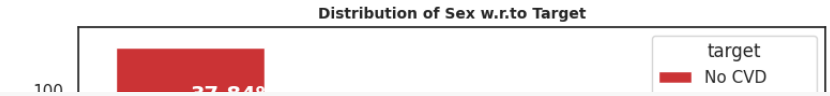
```
<ipython-input-151-b3b431164adb>:1: FutureWarning: The default value of numeric_only in DataFrame.skew is deprecated. In a future versic
  data.skew()
age        -0.214413
trestbps    0.710759
chol        1.129905
thalach    -0.531671
```

```
oldpeak     1.243552
ca          1.174624
dtype: float64
```

```
# count plot
sns.countplot(data=data, x='sex')
plt.show()
```



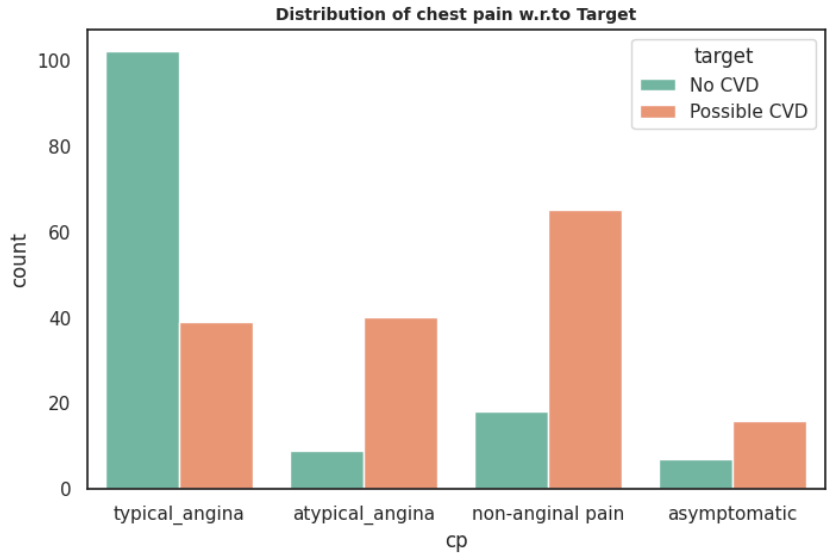

```
# Distribution of Gender by target
fig, ax = plt.subplots(figsize=(8, 5))

sns.countplot(x='sex', hue='target', data=data, palette='Set1', ax=ax)
ax.set_title("Distribution of Sex w.r.to Target", fontsize=10, weight='bold')
ax.set_xticklabels(['Female', 'Male'], rotation=0)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x() + i.get_width() / 2, i.get_height() - 15,
            f"{round((i.get_height() / total) * 100, 2)}%", fontsize=14,
            color='white', weight='bold')

plt.tight_layout()
plt.show()
```




```
# Distribution of Chest pain category based on

fig, ax = plt.subplots(figsize=(8, 5))

sns.countplot(x='cp', hue='target', data=data, palette='Set2', ax=ax)
ax.set_title("Distribution of chest pain w.r.to Target", fontsize=10, weight='bold')
ax.set_xticklabels(name, rotation=0)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x() + i.get_width() / 2, i.get_height() - 15,
            f"{round((i.get_height() / total) * 100, 2)}%", fontsize=14,
            color='white', weight='bold')

plt.tight_layout()
plt.show()
```

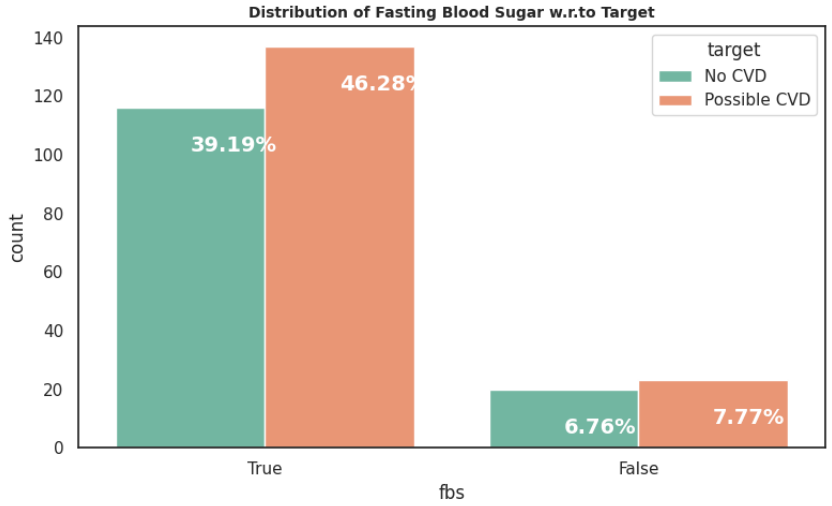
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-154-b3b080473176> in <cell line: 7>()
      5 sns.countplot(x='cp', hue='target', data=data, palette='Set2', ax=ax)
      6 ax.set_title("Distribution of chest pain w.r.to Target", fontsize=10,
weight='bold')
----> 7 ax.set_xticklabels(name, rotation=0)
      8
      9 totals = []

NameError: name 'name' is not defined
```

SEARCH STACK OVERFLOW




```
# Distribution of Fasting Blood sugar based on Target

fig, ax = plt.subplots(figsize=(8, 5))

sns.countplot(x='fbs', hue='target', data=data, palette='Set2', ax=ax)
ax.set_title("Distribution of Fasting Blood Sugar w.r.to Target", fontsize=10, weight='bold')
ax.set_xticklabels(['True','False'], rotation=0)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
```

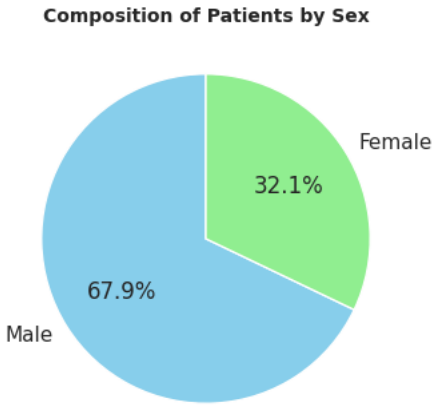
```
for i in ax.patches:
    ax.text(i.get_x() + i.get_width() / 2, i.get_height() - 15,
            f"{round((i.get_height() / total) * 100, 2)}%", fontsize=14,
            color='white', weight='bold')

plt.tight_layout()
plt.show()
```



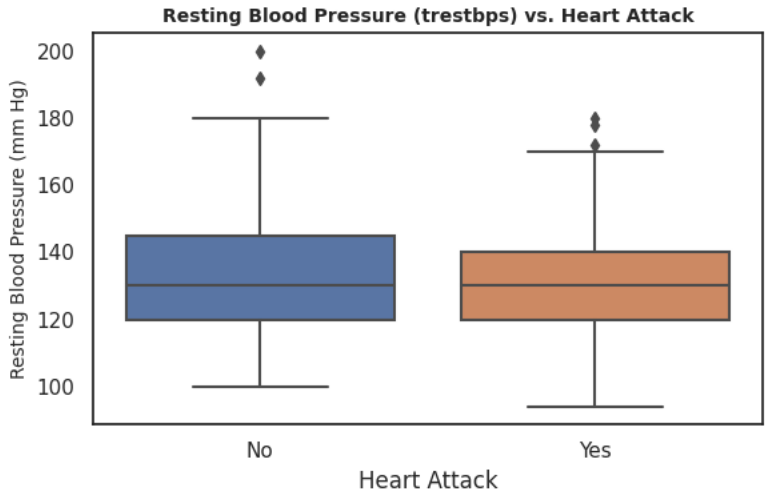

```
# Many patients with No CVD are found to have High Fasting Blood Sugar,
# so we can say that this is not a strong feature with respect to our target
```

```
# Distribution of Heart disease based on age category
fig, ax = plt.subplots(figsize=(10, 6))

sns.countplot(x='age', hue='target', data=data, palette='Set2', ax=ax)
ax.set_title("Distribution of CVD across age categories", fontsize=13, weight='bold')
ax.set_xlabel("Age", fontsize=10)
ax.set_ylabel("Count", fontsize=10)
ax.legend(title="CVD", labels=["No", "Yes"])

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x() + i.get_width() / 2, i.get_height() - 10,
            f"{round((i.get_height() / total) * 100, 2)}%", fontsize=10,
            color='white', weight='bold')

plt.tight_layout()
plt.show()
```

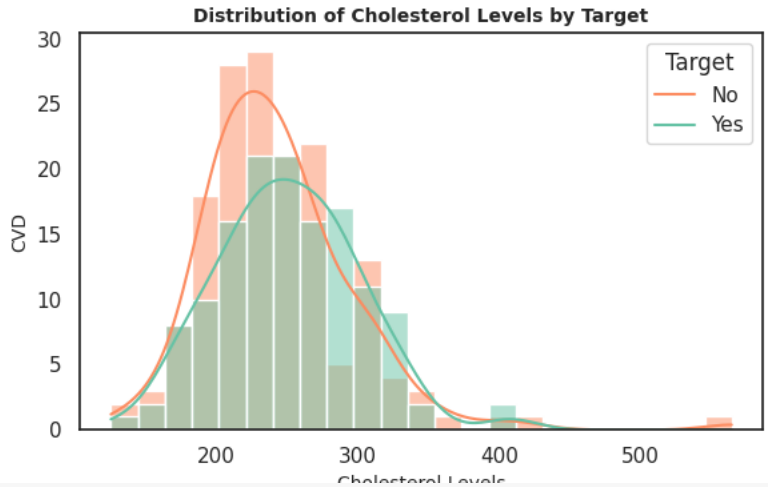
```
# Pie chart for gender composition based on the heart disease
sex_counts = data['sex'].value_counts()
labels = ['Male', 'Female']

fig, ax = plt.subplots(figsize=(4, 4))
ax.pie(sex_counts, labels=labels, autopct='%1.1f%%', startangle=90, colors=['skyblue', 'lightgreen'])
ax.set_title("Composition of Patients by Sex", fontsize=10, weight='bold')

plt.show()
```



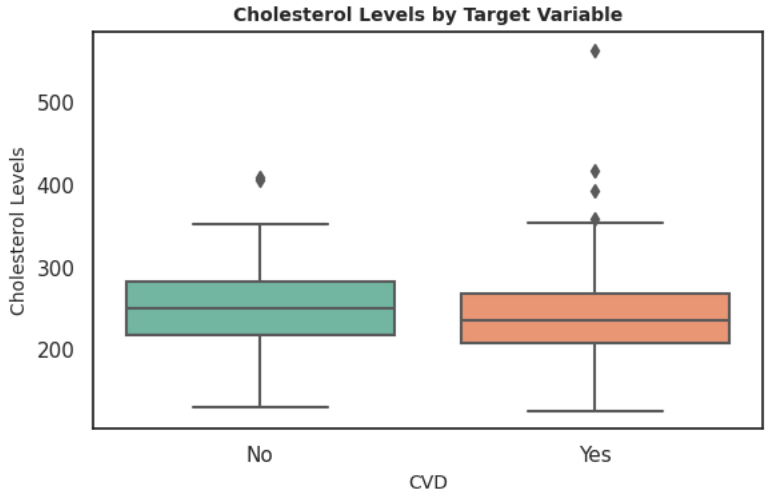

```
# Distribution of Resting Blood Pressure (trestbps) vs. Heart Attack
fig, ax = plt.subplots(figsize=(6, 4))
sns.boxplot(x='target', y='trestbps', data=data, ax=ax)
ax.set_title("Resting Blood Pressure (trestbps) vs. Heart Attack", fontsize=10, weight='bold')
ax.set_xlabel("Heart Attack", fontsize=12)
ax.set_ylabel("Resting Blood Pressure (mm Hg)", fontsize=10)
ax.set_xticklabels(["No", "Yes"])

plt.tight_layout()
plt.show()
```




```
# Distribution of Cholesterol Levels by Target
plt.figure(figsize=(6, 4))
sns.histplot(data=data, x='chol', hue='target', kde=True, palette='Set2')
plt.title("Distribution of Cholesterol Levels by Target", fontsize=10, weight='bold')
plt.xlabel("Cholesterol Levels", fontsize=10)
plt.ylabel("CVD", fontsize=10)
plt.legend(title="Target", labels=["No", "Yes"])

plt.tight_layout()
plt.show()
```

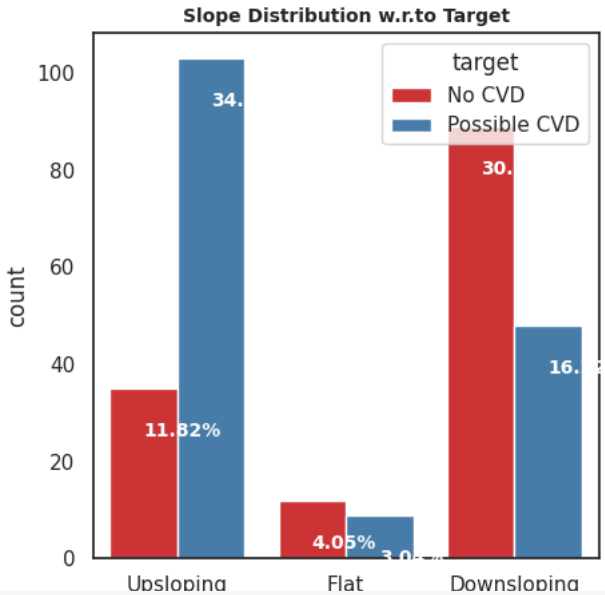



```
# Cholesterol Levels by Target Variable
plt.figure(figsize=(6, 4))
sns.boxplot(x='target', y='chol', data=data, palette='Set2')
plt.title("Cholesterol Levels by Target Variable", fontsize=10, weight='bold')
plt.xlabel("CVD", fontsize=10)
plt.ylabel("Cholesterol Levels", fontsize=10)
plt.xticks(ticks=[0, 1], labels=["No", "Yes"])

plt.tight_layout()
plt.show()
```

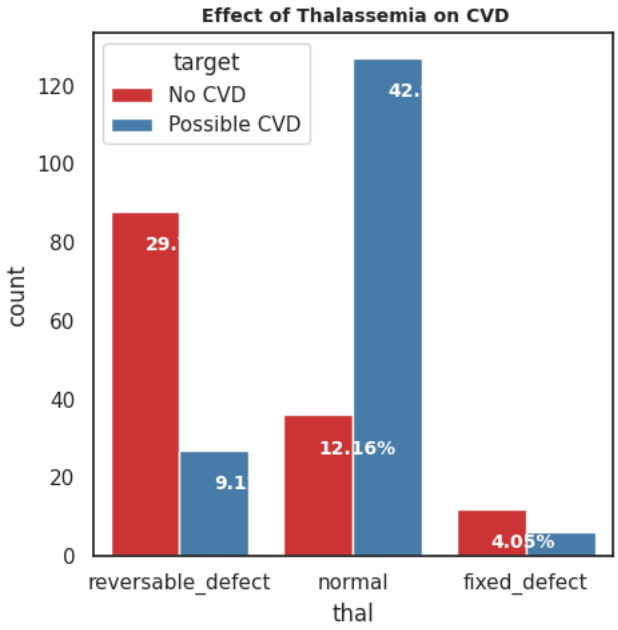



```
# We observe that The Cholesterol feature('chol') has less effect on the target
```

```
# Slope Distribution w.r.to Target
fig, ax = plt.subplots(figsize=(5, 5))

sns.countplot(x='slope', hue='target', data=data, palette='Set1', ax=ax)
ax.set_title("Slope Distribution w.r.to Target", fontsize=10, weight='bold')
ax.set_xticklabels(['Upsloping','Flat','Downsloping'], rotation=0)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x() + i.get_width() / 2, i.get_height() - 10,
            f"{round((i.get_height() / total) * 100, 2)}%", fontsize=10,
            color='white', weight='bold')

plt.tight_layout()
plt.show()
```




```
#Effect of Thalassemia on CVD

fig, ax = plt.subplots(figsize=(5, 5))

sns.countplot(x='thal', hue='target', data=data, palette='Set1', ax=ax)
ax.set_title(" Effect of Thalassemia on CVD", fontsize=10, weight='bold')
ax.set_xticklabels(['reversable_defect','normal','fixed_defect'], rotation=0)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x() + i.get_width() / 2, i.get_height() - 10,
            f"{round((i.get_height() / total) * 100, 2)}%", fontsize=10,
            color='white', weight='bold')

plt.tight_layout()
plt.show()
```




```
data.to_csv('cleaned_dataHeartdiseaseDA.csv', index=False)
```