Calculation of mean, variance, skew, kurtosis for the datasets

[9]:
```
# calculate mean
df_heart[choosen_features_nums].mean()
```

[9]:
```
oldpeak      1.113106
thalach    149.647978
dtype: float64
```

[10]:
```
# calculate variance
df_heart[choosen_features_nums].var()
```

[10]:
```
oldpeak      1.577304
thalach    487.358850
dtype: float64
```

[11]:
```
# calculate skew
df_heart[choosen_features_nums].skew()
```

[11]:
```
oldpeak    1.224053
thalach   -0.394100
dtype: float64
```

[12]:
```
# calculate kurtosis
df_heart[choosen_features_nums].kurtosis()
```

[12]:
```
oldpeak    1.363172
thalach   -0.214108
dtype: float64
```

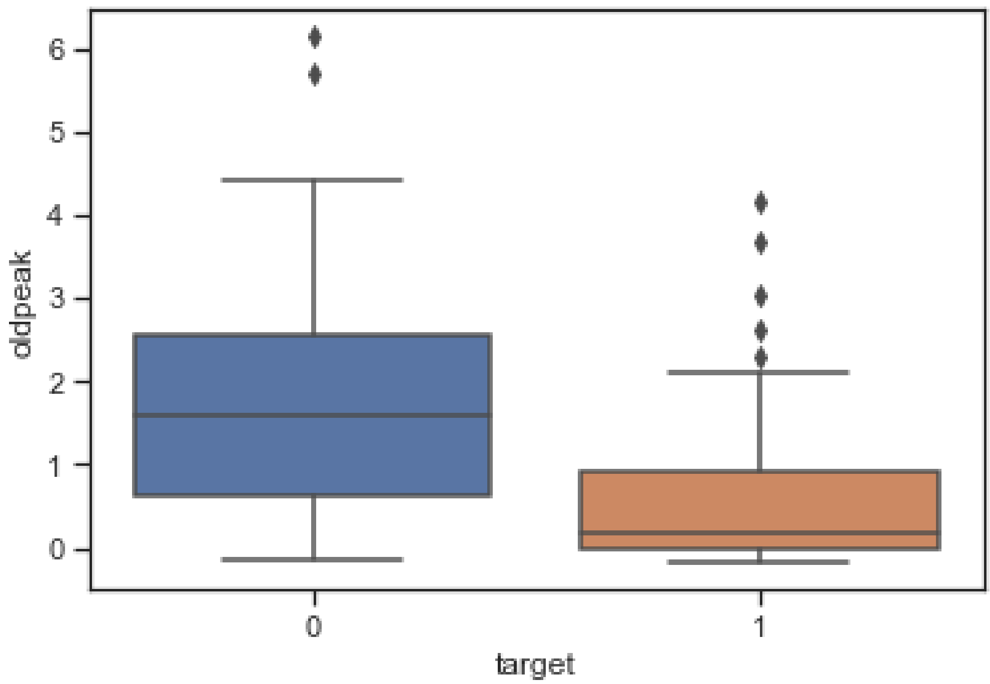
From mean, var, skew, kurtosis, we observe that:

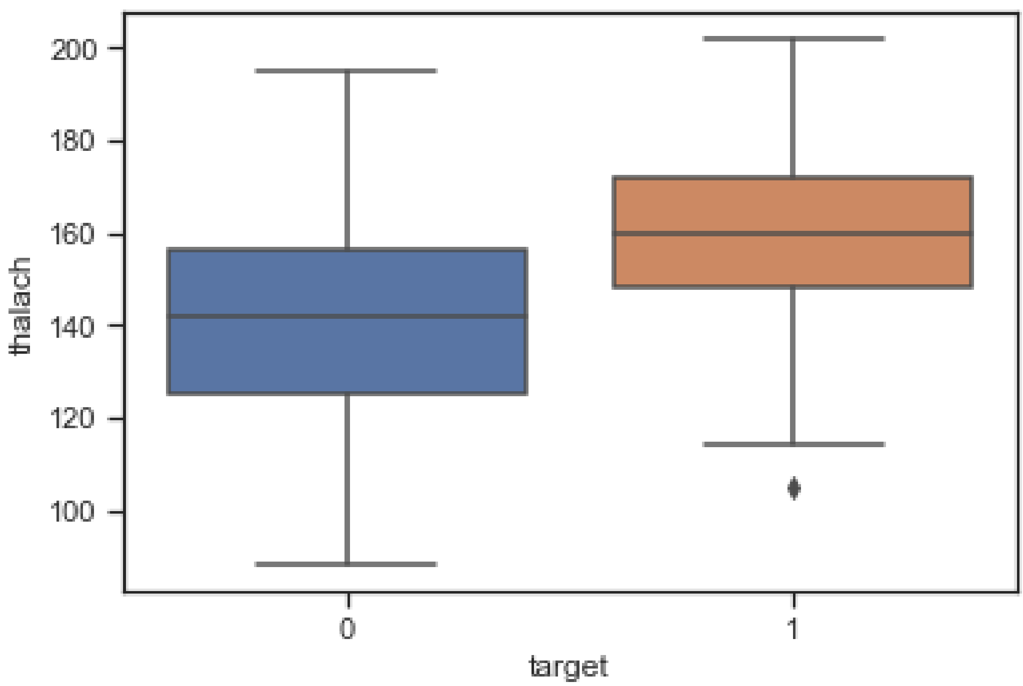
- oldpeak is left-skewed, whereas thalach is right skewed.
- the kurtosis values for thalach is negative indicating light tail distribution. Whereas, oldpeak has positive kurtosis value indicating heavy tail distribution.
- variance for oldpeak and thalach are high indicating that the values are highly spread out from the mean.

[CM3]

Checking for notable outliers using "Box Plots"

[13]:
```
# boxplot for outlier detection of numerical features
for column in df_heart[choosen_features_nums]:
    plt.figure()
    ax = sns.boxplot(x='target', y=column, data=df_heart)
    plt.show()
```





From the above box-plot of selected numerical features 'oldpeak' & 'thalach': - we notice few outliers in oldpeak - an outlier in thalach

Checking for outliers using IQR

[14]:
```
# outlier detection using IQR
for column in df_heart[choosen_features_nums]:
    for target in df_heart['target'].unique():
        q25 = df_heart[column][df_heart['target'] == target].quantile(0.25)
        q75 = df_heart[column][df_heart['target'] == target].quantile(0.75)
        iqr = q75 - q25
        print(target, '-', column.upper())
        print('Percentiles: 25th = %.3f, 75th = %.3f, IQR = %.3f' % (q25, q75, iqr))

        # Calculate the outlier cutoff
        cut_off = iqr * 1.5
        lower, upper = q25 - cut_off, q75 + cut_off

        # Identify outliers
        df_heart2 = pd.DataFrame(df_heart[df_heart['target'] == target][column])

        count = len(df_heart2[df_heart2[column] < lower].index)
        count += len(df_heart2[df_heart2[column] > upper].index)
        print('Identified outliers: ', count)

        # replacing outliers with NaN (Will be later replaced with feature mean)
        for index in df_heart2[df_heart2[column] < lower].index:
            df_heart.loc[index, column] = np.nan
        for index in df_heart2[df_heart2[column] > upper].index:
            df_heart.loc[index, column] = np.nan
```

```
1 - OLDPEAK
Percentiles: 25th = -0.023, 75th = 0.906, IQR = 0.929
Identified outliers:  5
0 - OLDPEAK
Percentiles: 25th = 0.623, 75th = 2.555, IQR = 1.932
Identified outliers:  2
1 - THALACH
Percentiles: 25th = 148.052, 75th = 172.048, IQR = 23.996
Identified outliers:  1
0 - THALACH
Percentiles: 25th = 124.972, 75th = 156.158, IQR = 31.186
Identified outliers:  0
```

We observe that the number of outliers found corresponds to the box-plot. These outliers can be handled by replacing with feature mean.

[CM4]

Histogram plot of the features

[15]:
```
# plot histogram
for column in df_heart[choosen_features]:
    sns.displot(df_heart, x=column, hue="target")
```