

# Data Analysis Capstone

January 18, 2022

```
[1]: import pandas as pd
import numpy as np
```

```
[2]: df=pd.read_excel("data.xlsx")
print(df)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	63	1	3	145	233	1	0	150	0	2.3	
1	37	1	2	130	250	0	1	187	0	3.5	
2	41	0	1	130	204	0	0	172	0	1.4	
3	56	1	1	120	236	0	1	178	0	0.8	
4	57	0	0	120	354	0	1	163	1	0.6	
..	...	...	..	...	...	...	...	...	...	...	
298	57	0	0	140	241	0	1	123	1	0.2	
299	45	1	3	110	264	0	1	132	0	1.2	
300	68	1	0	144	193	1	1	141	0	3.4	
301	57	1	0	130	131	0	1	115	1	1.2	
302	57	0	1	130	236	0	0	174	0	0.0	

	slope	ca	thal	target
0	0	0	1	1
1	0	0	2	1
2	2	0	2	1
3	2	0	2	1
4	2	0	2	1
..	...	..	...	...
298	1	0	3	0
299	1	0	3	0
300	1	2	3	0
301	1	1	3	0
302	1	1	2	0

[303 rows x 14 columns]

```
[3]: df.isnull()
```

```
[3]:      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  \
0  False False False      False False False      False  False  False
```

1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
..	...	...	...	...	...	...	...	...	...
298	False	False	False	False	False	False	False	False	False
299	False	False	False	False	False	False	False	False	False
300	False	False	False	False	False	False	False	False	False
301	False	False	False	False	False	False	False	False	False
302	False	False	False	False	False	False	False	False	False

	oldpeak	slope	ca	thal	target
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
..	...	...	...	...	...
298	False	False	False	False	False
299	False	False	False	False	False
300	False	False	False	False	False
301	False	False	False	False	False
302	False	False	False	False	False

[303 rows x 14 columns]

```
[4]: df.isnull().sum()
```

```
[4]: 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
```

```
[5]: # There are no missing values in the data
```

```
[6]: duplicate = df[df.duplicated()]
print("Duplicate Rows :")
```

Duplicate Rows :

```
[7]: duplicate
```

```
[7]:      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
164   38    1   2      138   175    0         1      173     0       0.0

      slope  ca  thal  target
164      2   4     2       1
```

```
[8]: df.duplicated()
```

```
[8]: 0      False
1      False
2      False
3      False
4      False
...
298    False
299    False
300    False
301    False
302    False
Length: 303, dtype: bool
```

```
[9]: df.duplicated().sum()
```

```
[9]: 1
```

```
[10]: df=df.drop_duplicates()
```

```
[11]: df
```

```
[11]:      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0      63    1   3      145   233    1         0      150     0       2.3
1      37    1   2      130   250    0         1      187     0       3.5
2      41    0   1      130   204    0         0      172     0       1.4
3      56    1   1      120   236    0         1      178     0       0.8
4      57    0   0      120   354    0         1      163     1       0.6
...  ...  ...  ..      ...  ...  ...      ...  ...  ...
298   57    0   0      140   241    0         1      123     1       0.2
299   45    1   3      110   264    0         1      132     0       1.2
300   68    1   0      144   193    1         1      141     0       3.4
301   57    1   0      130   131    0         1      115     1       1.2
302   57    0   1      130   236    0         0      174     0       0.0
```

	slope	ca	thal	target
0	0	0	1	1
1	0	0	2	1
2	2	0	2	1
3	2	0	2	1
4	2	0	2	1
..	...	..	...	...
298	1	0	3	0
299	1	0	3	0
300	1	2	3	0
301	1	1	3	0
302	1	1	2	0

[302 rows x 14 columns]

```
[12]: # One duplicate removed from the data set
```

```
[13]: df.mean()
```

```
[13]: age          54.420530
sex            0.682119
cp             0.963576
trestbps      131.602649
chol          246.500000
fbs           0.149007
restecg       0.526490
thalach       149.569536
exang         0.327815
oldpeak       1.043046
slope         1.397351
ca            0.718543
thal         2.314570
target        0.543046
dtype: float64
```

```
[14]: df.mode()
```

```
[14]:   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0  58.0  1.0  0.0    120.0   197  0.0      1.0    162.0   0.0     0.0
1   NaN  NaN  NaN      NaN   204  NaN     NaN      NaN   NaN     NaN
2   NaN  NaN  NaN      NaN   234  NaN     NaN      NaN   NaN     NaN

   slope  ca  thal  target
0    2.0  0.0  2.0     1.0
1   NaN  NaN  NaN     NaN
2   NaN  NaN  NaN     NaN
```

```
[15]: df.median()
```

```
[15]: age          55.5  
sex            1.0  
cp             1.0  
trestbps      130.0  
chol          240.5  
fbs           0.0  
restecg       1.0  
thalach       152.5  
exang         0.0  
oldpeak       0.8  
slope         1.0  
ca            0.0  
thal          2.0  
target        1.0  
dtype: float64
```

```
[16]: import statistics
```

```
[17]: df.std()
```

```
[17]: age          9.047970  
sex          0.466426  
cp           1.032044  
trestbps     17.563394  
chol         51.753489  
fbs          0.356686  
restecg      0.526027  
thalach      22.903527  
exang        0.470196  
oldpeak      1.161452  
slope        0.616274  
ca           1.006748  
thal         0.613026  
target       0.498970  
dtype: float64
```

```
[18]: df.describe()
```

```
[18]:
```

	age	sex	cp	trestbps	chol	fbs \
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000

75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000

	restecg	thalach	exang	oldpeak	slope	ca \
count	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543
std	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748
min	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	133.250000	0.000000	0.000000	1.000000	0.000000
50%	1.000000	152.500000	0.000000	0.800000	1.000000	0.000000
75%	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000
max	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000

	thal	target
count	302.000000	302.000000
mean	2.314570	0.543046
std	0.613026	0.498970
min	0.000000	0.000000
25%	2.000000	0.000000
50%	2.000000	1.000000
75%	3.000000	1.000000
max	3.000000	1.000000

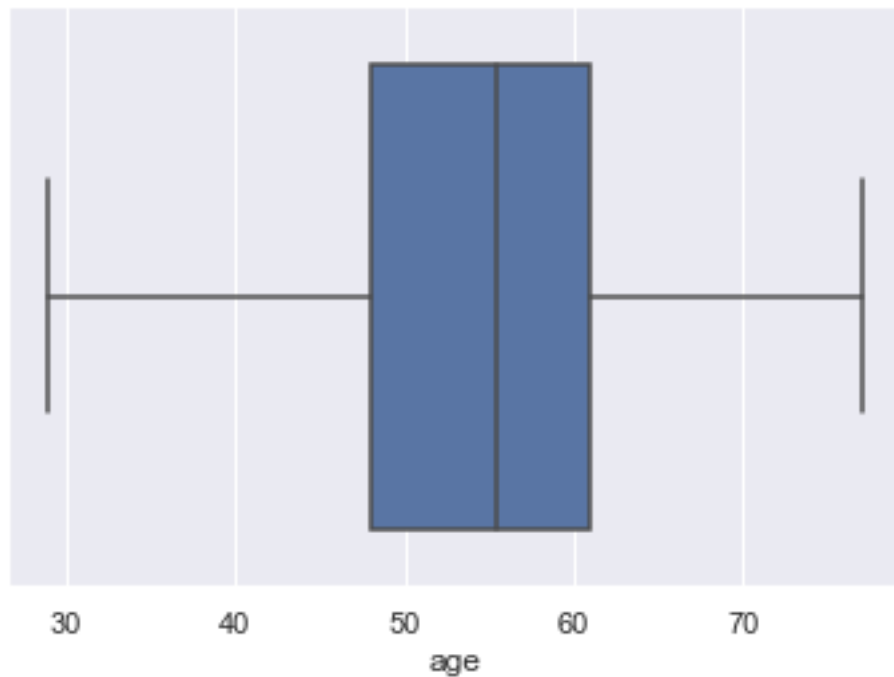
```
[19]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 302 entries, 0 to 302
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  -
0   age         302 non-null    int64
1   sex         302 non-null    int64
2   cp          302 non-null    int64
3   trestbps    302 non-null    int64
4   chol        302 non-null    int64
5   fbs         302 non-null    int64
6   restecg     302 non-null    int64
7   thalach     302 non-null    int64
8   exang       302 non-null    int64
9   oldpeak     302 non-null    float64
10  slope       302 non-null    int64
11  ca          302 non-null    int64
12  thal        302 non-null    int64
13  target      302 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 35.4 KB
```

```
[20]: import seaborn as sns #visualisation
import matplotlib.pyplot as plt #visualisation
%matplotlib inline
sns.set(color_codes=True)
```

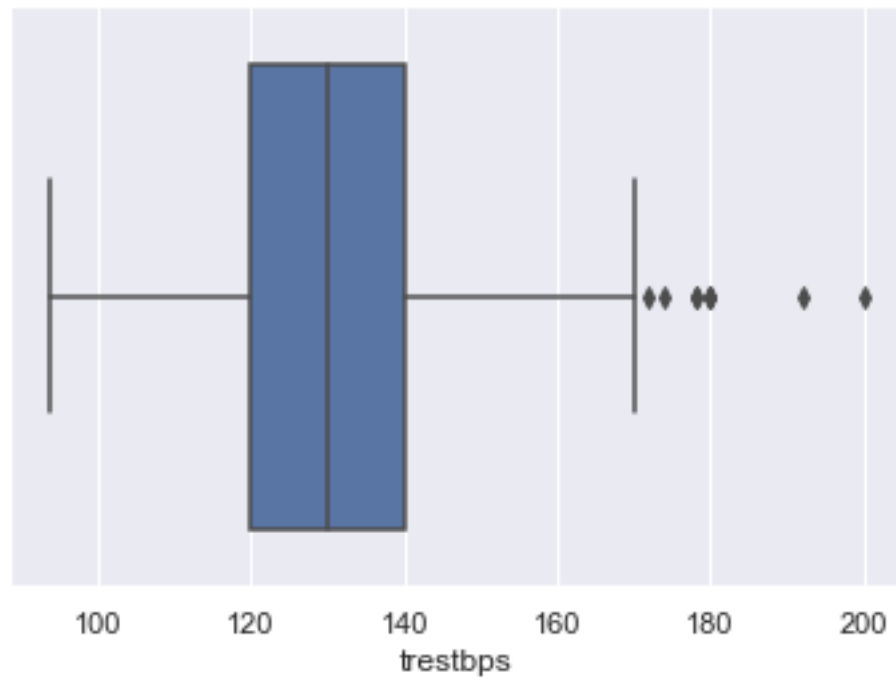
```
[21]: sns.boxplot(x=df['age'])
```

```
[21]: <AxesSubplot:xlabel='age'>
```



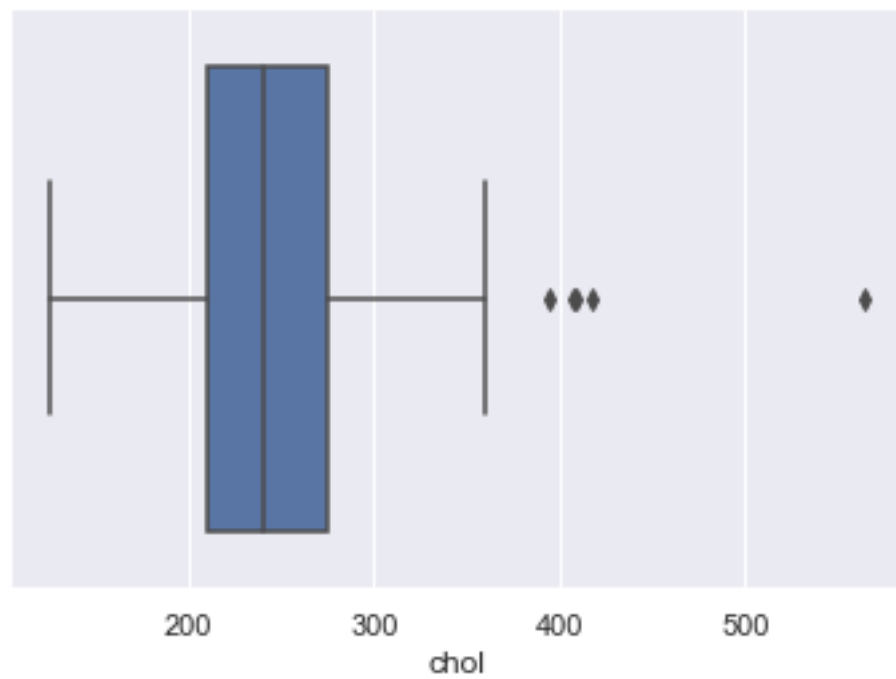
```
[22]: sns.boxplot(x=df['trestbps'])
```

```
[22]: <AxesSubplot:xlabel='trestbps'>
```



```
[23]: sns.boxplot(x=df['chol'])
```

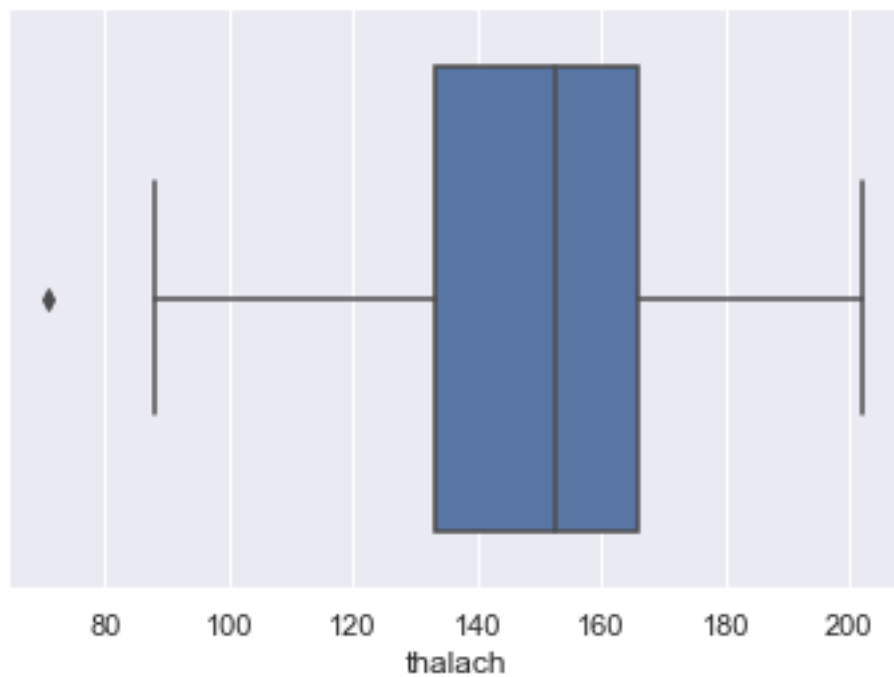
```
[23]: <AxesSubplot:xlabel='chol'>
```





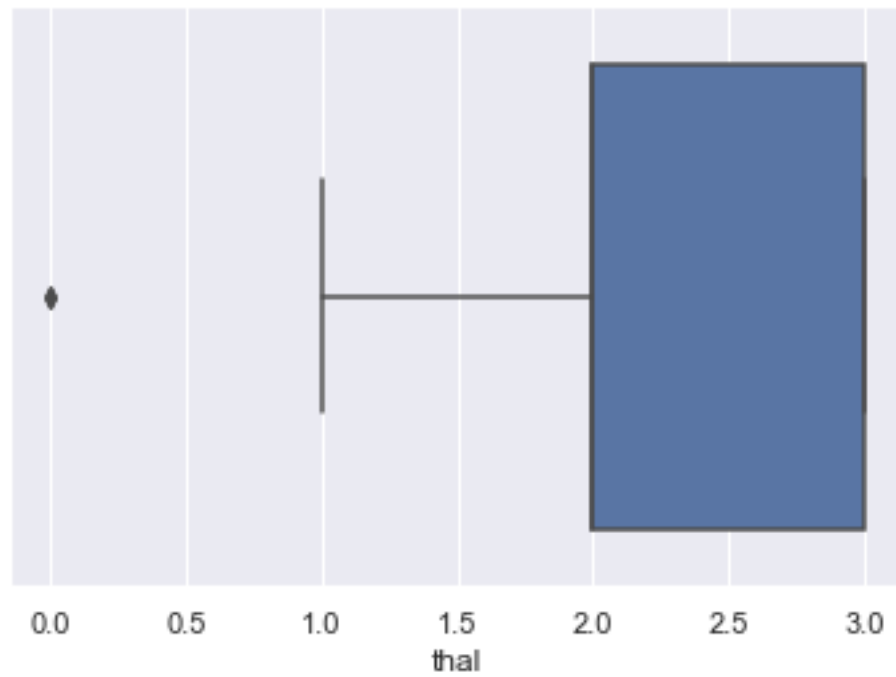
```
[24]: sns.boxplot(x=df['thalach'])
```

```
[24]: <AxesSubplot:xlabel='thalach'>
```



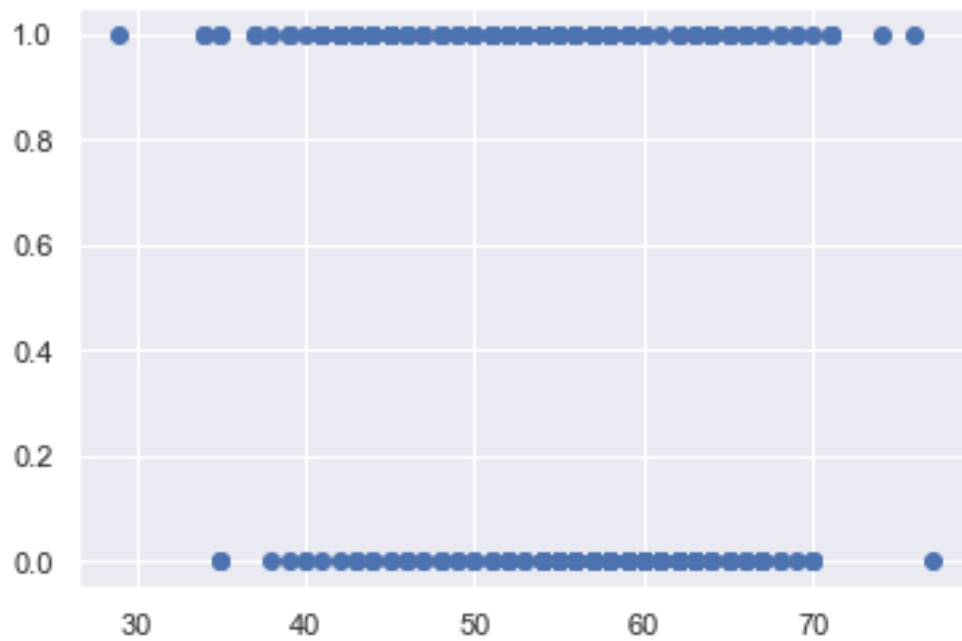
```
[25]: sns.boxplot(x=df['thal'])
```

```
[25]: <AxesSubplot:xlabel='thal'>
```



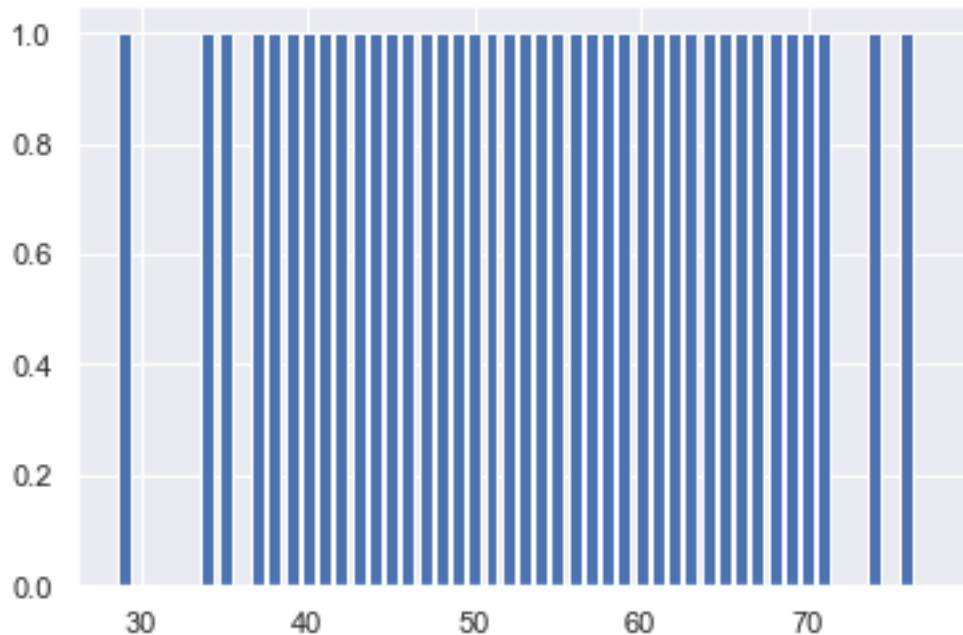
```
[28]: plt.scatter(df['age'], df['target'])
```

```
[28]: <matplotlib.collections.PathCollection at 0x2136b1f9eb0>
```



```
[29]: plt.bar(df['age'], df['target'])
```

```
[29]: <BarContainer object of 302 artists>
```



```
[30]: categorical_values = []
for column in df.columns:
    print('=====')
    print(f"{column} : {df[column].unique()}")
    if len(df[column].unique()) <= 10:
        categorical_values.append(column)
```

```
=====
age : [63 37 41 56 57 44 52 54 48 49 64 58 50 66 43 69 59 42 61 40 71 51 65 53
      46 45 39 47 62 34 35 29 55 60 67 68 74 76 70 38 77]
=====
sex : [1 0]
=====
cp : [3 2 1 0]
=====
trestbps : [145 130 120 140 172 150 110 135 160 105 125 142 155 104 138 128 108
            134
            122 115 118 100 124  94 112 102 152 101 132 148 178 129 180 136 126 106
            156 170 146 117 200 165 174 192 144 123 154 114 164]
=====
chol : [233 250 204 236 354 192 294 263 199 168 239 275 266 211 283 219 340 226
        247 234 243 302 212 175 417 197 198 177 273 213 304 232 269 360 308 245]
```

```

208 264 321 325 235 257 216 256 231 141 252 201 222 260 182 303 265 309
186 203 183 220 209 258 227 261 221 205 240 318 298 564 277 214 248 255
207 223 288 160 394 315 246 244 270 195 196 254 126 313 262 215 193 271
268 267 210 295 306 178 242 180 228 149 278 253 342 157 286 229 284 224
206 167 230 335 276 353 225 330 290 172 305 188 282 185 326 274 164 307
249 341 407 217 174 281 289 322 299 300 293 184 409 259 200 327 237 218
319 166 311 169 187 176 241 131]
=====
fbs : [1 0]
=====
restecg : [0 1 2]
=====
thalach : [150 187 172 178 163 148 153 173 162 174 160 139 171 144 158 114 151
161
179 137 157 123 152 168 140 188 125 170 165 142 180 143 182 156 115 149
146 175 186 185 159 130 190 132 147 154 202 166 164 184 122 169 138 111
145 194 131 133 155 167 192 121 96 126 105 181 116 108 129 120 112 128
109 113 99 177 141 136 97 127 103 124 88 195 106 95 117 71 118 134
90]
=====
exang : [0 1]
=====
oldpeak : [2.3 3.5 1.4 0.8 0.6 0.4 1.3 0. 0.5 1.6 1.2 0.2 1.8 1. 2.6 1.5 3.
2.4
0.1 1.9 4.2 1.1 2. 0.7 0.3 0.9 3.6 3.1 3.2 2.5 2.2 2.8 3.4 6.2 4. 5.6
2.9 2.1 3.8 4.4]
=====
slope : [0 2 1]
=====
ca : [0 2 1 3 4]
=====
thal : [1 2 3 0]
=====
target : [1 0]

```

```
[31]: categorical_values
```

```
[31]: ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

```
[32]: df.describe()
```

```
[32]:
```

	age	sex	cp	trestbps	chol	fbs \
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000

75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000

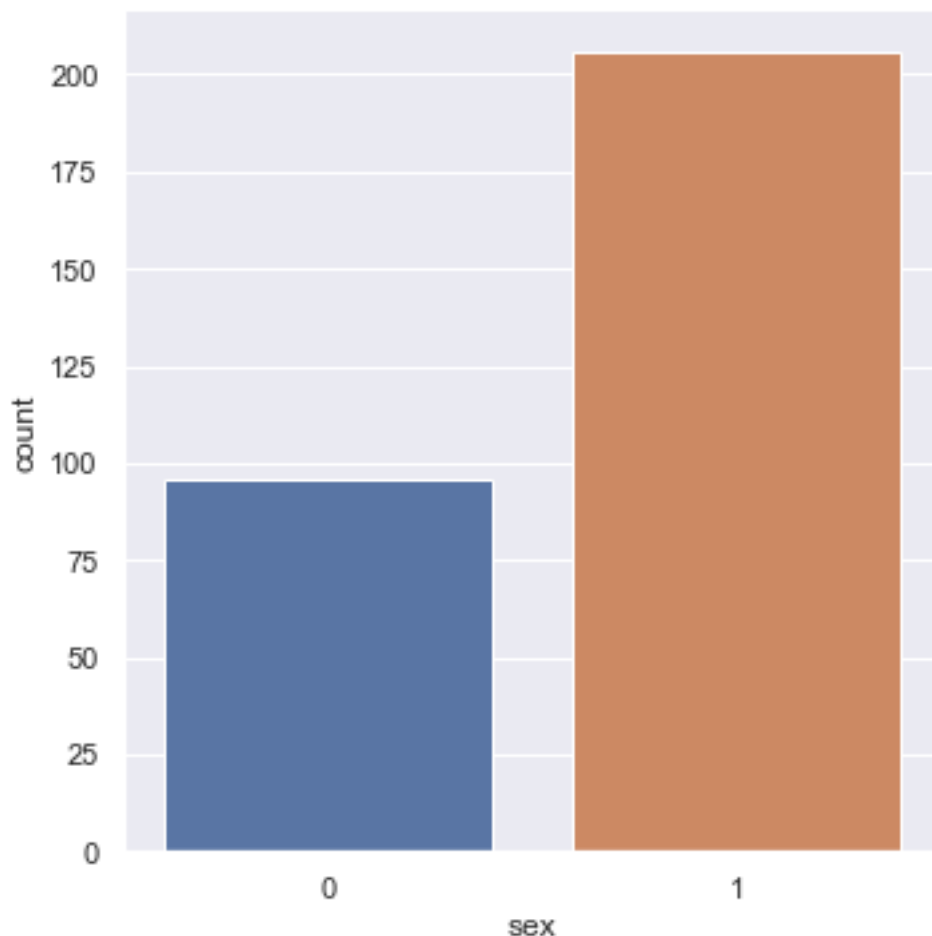
	restecg	thalach	exang	oldpeak	slope	ca \
count	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543
std	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748
min	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	133.250000	0.000000	0.000000	1.000000	0.000000
50%	1.000000	152.500000	0.000000	0.800000	1.000000	0.000000
75%	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000
max	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000

	thal	target
count	302.000000	302.000000
mean	2.314570	0.543046
std	0.613026	0.498970
min	0.000000	0.000000
25%	2.000000	0.000000
50%	2.000000	1.000000
75%	3.000000	1.000000
max	3.000000	1.000000

```
[33]: sns.factorplot('sex', data=df, kind='count')
```

```
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\categorical.py:3704:
UserWarning: The `factorplot` function has been renamed to `catplot`. The
original name will be removed in a future release. Please update your code. Note
that the default `kind` in `factorplot` (`'point'`) has changed to `strip` in
`catplot`.
  warnings.warn(msg)
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  warnings.warn(
```

```
[33]: <seaborn.axisgrid.FacetGrid at 0x2136aea9c70>
```



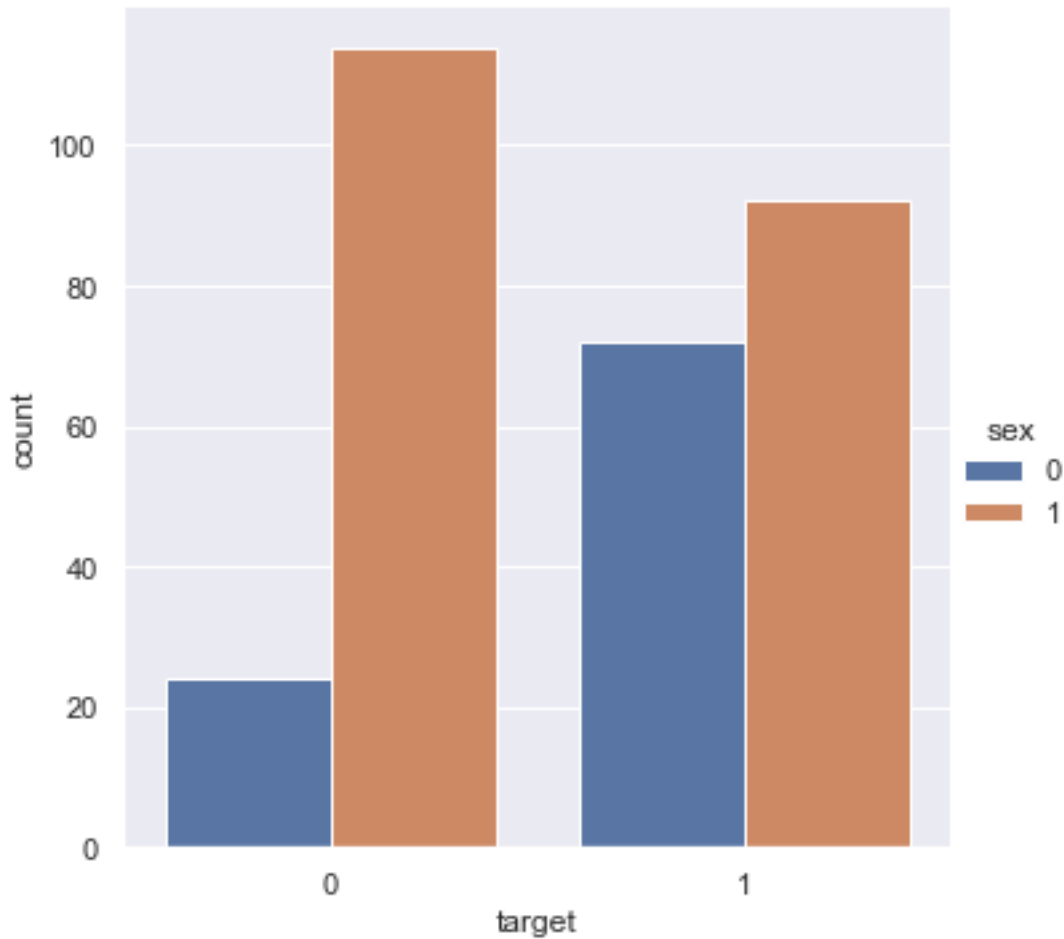
```
[34]: sns.factorplot('target', data=df, hue='sex', kind='count')
```

```
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\categorical.py:3704:
UserWarning: The `factorplot` function has been renamed to `catplot`. The
original name will be removed in a future release. Please update your code. Note
that the default `kind` in `factorplot` (`'point'`) has changed to `strip` in
`catplot`.
```

```
warnings.warn(msg)
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
```

```
warnings.warn(
```

```
[34]: <seaborn.axisgrid.FacetGrid at 0x2136b2a4670>
```

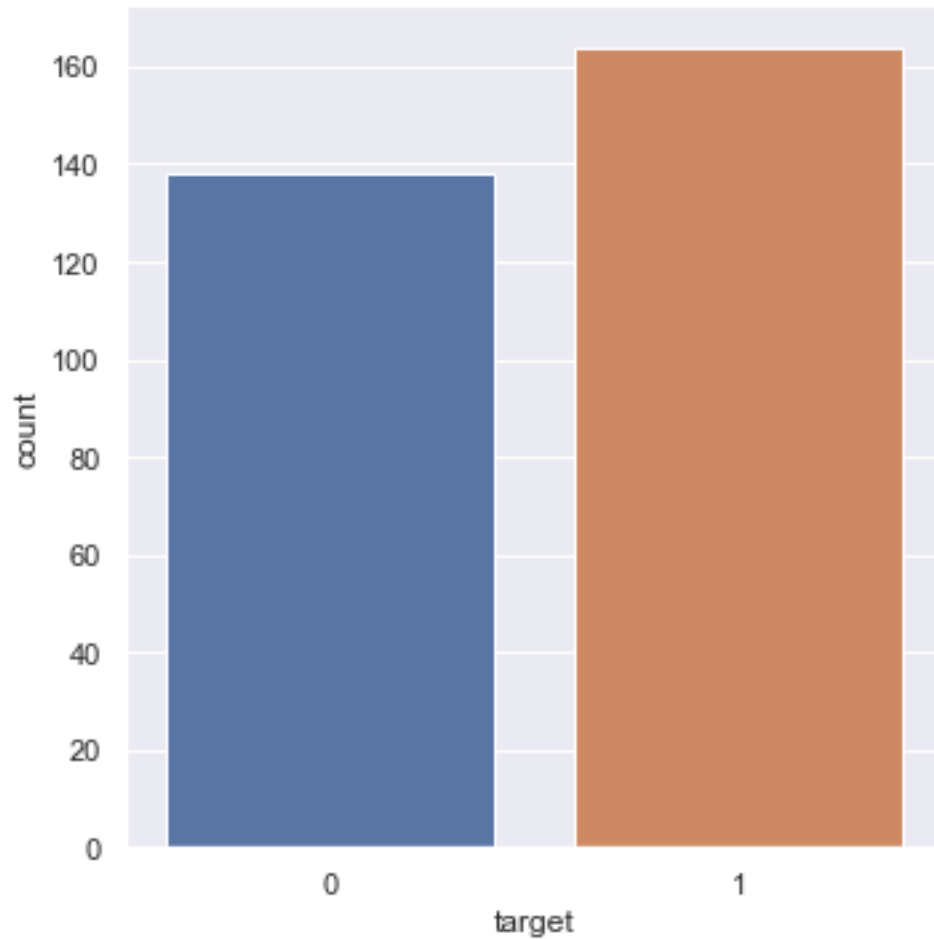


```
[35]: sns.factorplot('target', data=df, kind='count')
```

C:\Users\nicme\anaconda3\lib\site-packages\seaborn\categorical.py:3704:  
 UserWarning: The `factorplot` function has been renamed to `catplot`. The original name will be removed in a future release. Please update your code. Note that the default `kind` in `factorplot` (`'point'`) has changed to `strip` in `catplot`.

warnings.warn(msg)  
 C:\Users\nicme\anaconda3\lib\site-packages\seaborn\\_decorators.py:36:  
 FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.  
 warnings.warn(

```
[35]: <seaborn.axisgrid.FacetGrid at 0x2136b6a1f10>
```



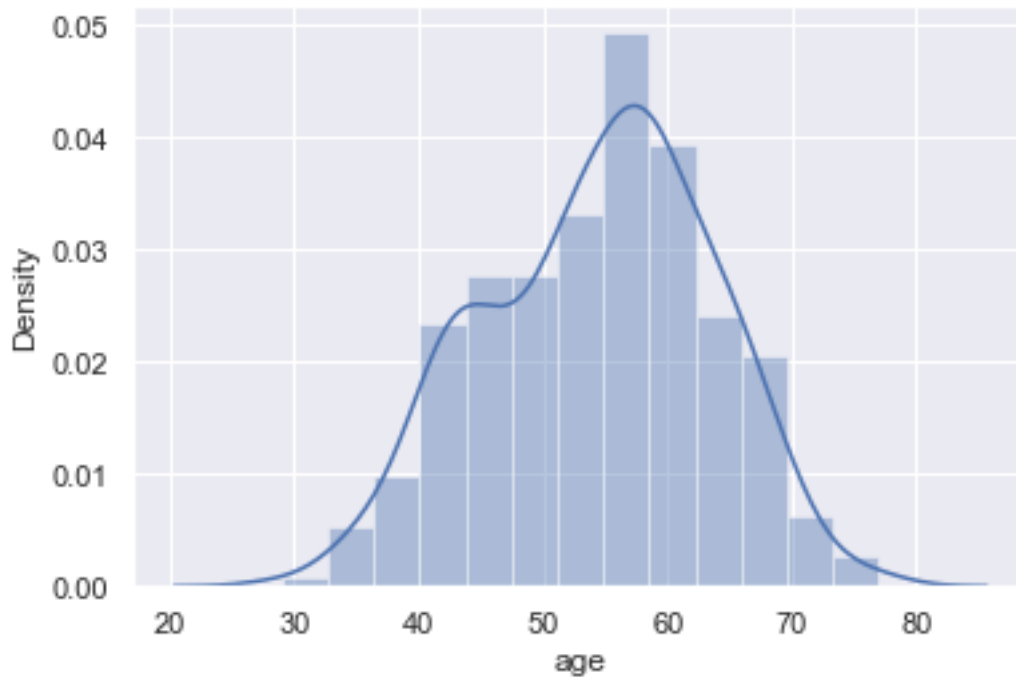
```
[41]: sns.distplot(df['age'])
```

```
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\distributions.py:2551:  
FutureWarning: `distplot` is a deprecated function and will be removed in a  
future version. Please adapt your code to use either `displot` (a figure-level  
function with similar flexibility) or `histplot` (an axes-level function for  
histograms).
```

```
warnings.warn(msg, FutureWarning)
```

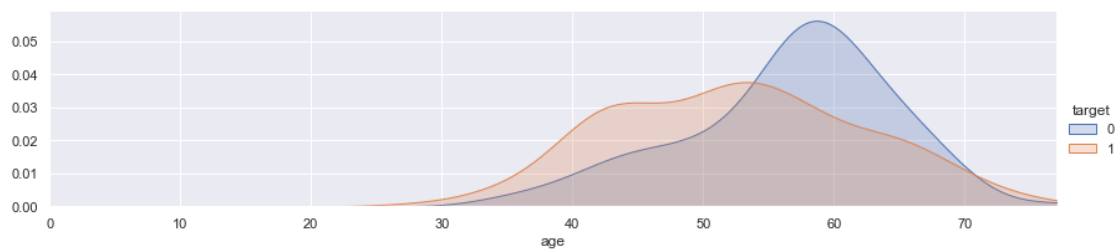
```
[41]: <AxesSubplot:xlabel='age', ylabel='Density'>
```





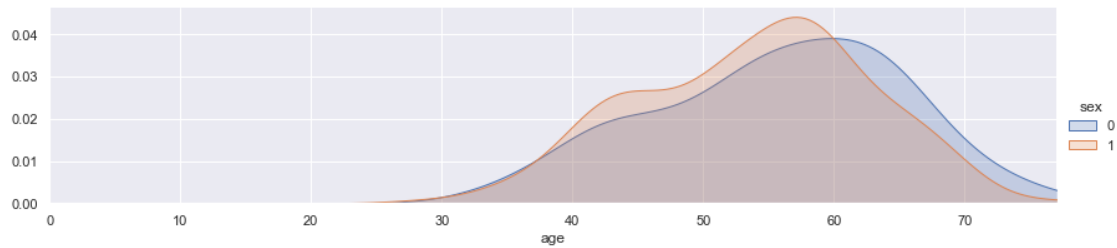
```
[38]: fig = sns.FacetGrid(df, hue="target", aspect=4)
fig.map(sns.kdeplot, 'age', shade=True)
oldest = df['age'].max()
fig.set(xlim=(0, oldest))
fig.add_legend()
```

[38]: <seaborn.axisgrid.FacetGrid at 0x2136bb5ac70>



```
[39]: fig = sns.FacetGrid(df, hue="sex", aspect=4)
fig.map(sns.kdeplot, 'age', shade=True)
oldest = df['age'].max()
fig.set(xlim=(0, oldest))
fig.add_legend()
```

[39]: <seaborn.axisgrid.FacetGrid at 0x2136b7e5fa0>

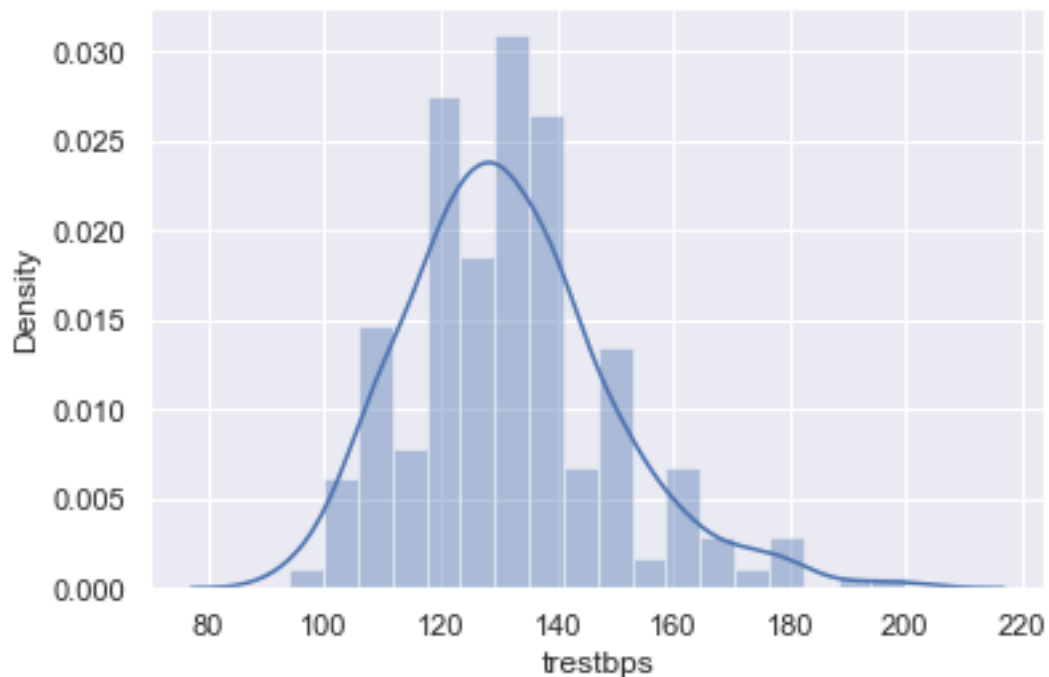


[40]: `sns.distplot(df['trestbps'])`

C:\Users\nicme\anaconda3\lib\site-packages\seaborn\distributions.py:2551:  
FutureWarning: `distplot` is a deprecated function and will be removed in a  
future version. Please adapt your code to use either `displot` (a figure-level  
function with similar flexibility) or `histplot` (an axes-level function for  
histograms).

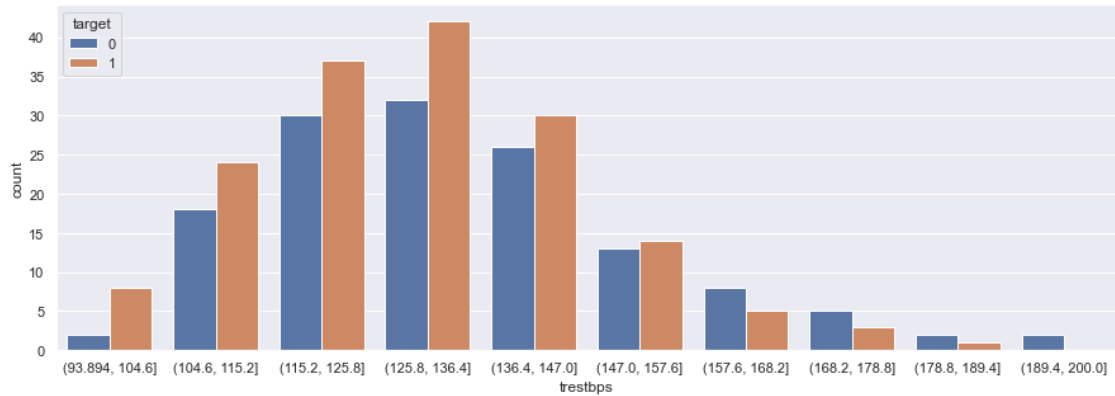
warnings.warn(msg, FutureWarning)

[40]: <AxesSubplot:xlabel='trestbps', ylabel='Density'>



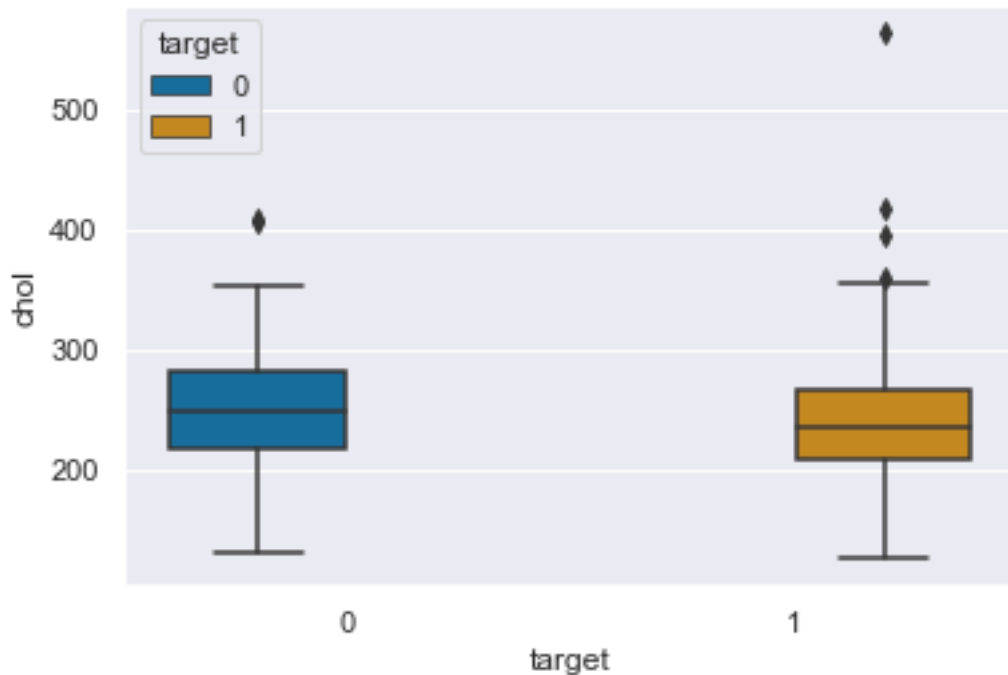
```
[43]: f, ax = plt.subplots(figsize=(15, 5))
sns.countplot(data=df, x=pd.cut(df['trestbps'], 10), hue='target')
```

```
[43]: <AxesSubplot:xlabel='trestbps', ylabel='count'>
```



```
[47]: sns.boxplot(y='chol', x='target',
                  data=df,
                  palette="colorblind",
                  hue='target')
```

```
[47]: <AxesSubplot:xlabel='target', ylabel='chol'>
```



```
[48]: # there seems to be an even distributution between our target and cholesterol_
      ↪ variables. Those with CVD have more outliers than those without.
```

```
[50]: sns.factorplot('slope', data=df, hue='target', kind='count')
```

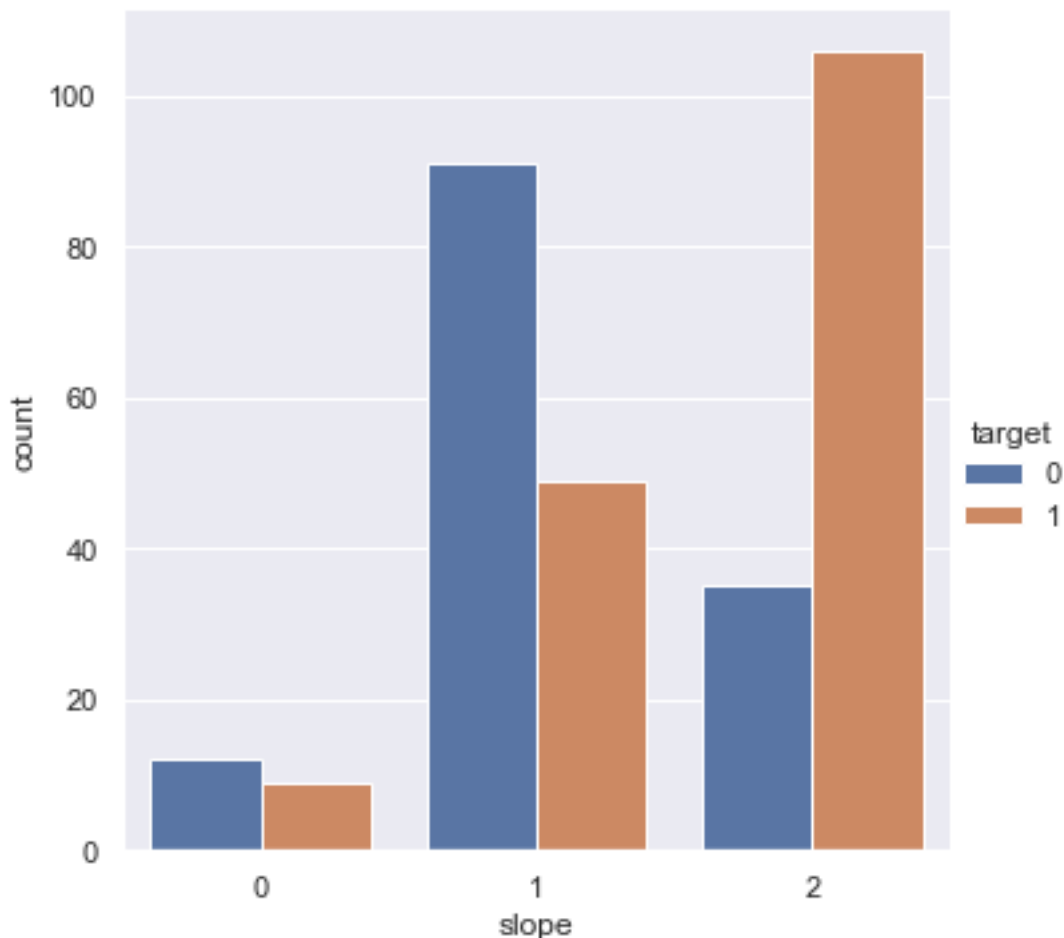
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\categorical.py:3704:  
UserWarning: The `factorplot` function has been renamed to `catplot`. The original name will be removed in a future release. Please update your code. Note that the default `kind` in `factorplot` (`'point'`) has changed to `strip` in `catplot`.

```
warnings.warn(msg)
```

C:\Users\nicme\anaconda3\lib\site-packages\seaborn\\_decorators.py:36:  
FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

```
warnings.warn(
```

```
[50]: <seaborn.axisgrid.FacetGrid at 0x2136cfc87c0>
```



```
[51]: sns.factorplot('exang', data=df, hue='target', kind='count')
```

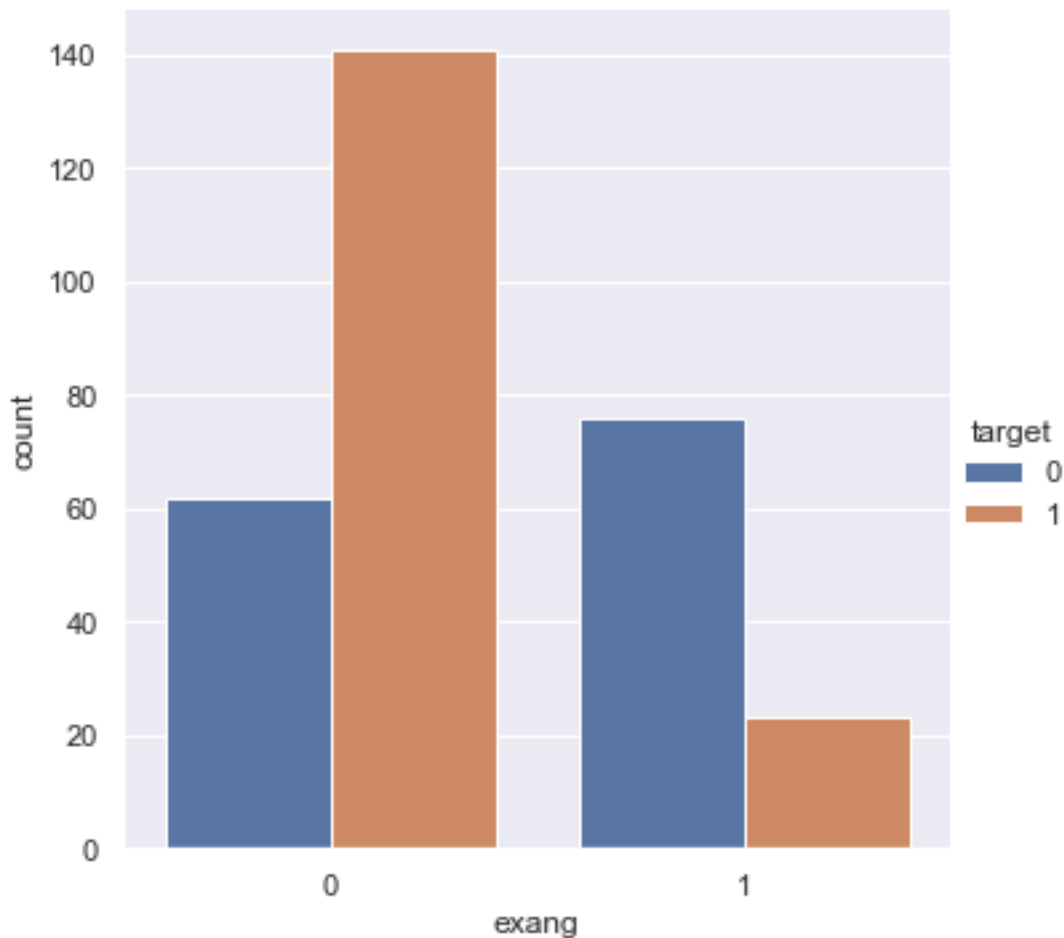
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\categorical.py:3704:  
UserWarning: The `factorplot` function has been renamed to `catplot`. The  
original name will be removed in a future release. Please update your code. Note  
that the default `kind` in `factorplot` (`'point'`) has changed to `strip` in  
`catplot`.

```
warnings.warn(msg)
```

C:\Users\nicme\anaconda3\lib\site-packages\seaborn\\_decorators.py:36:  
FutureWarning: Pass the following variable as a keyword arg: x. From version  
0.12, the only valid positional argument will be `data`, and passing other  
arguments without an explicit keyword will result in an error or  
misinterpretation.

```
warnings.warn(
```

```
[51]: <seaborn.axisgrid.FacetGrid at 0x2136d01cf70>
```



```
[52]: # People with a downslope during peak exercise are at a greater risk for CVD.
```

```
[53]: sns.factorplot('thal', data=df, hue='target', kind='count')
```

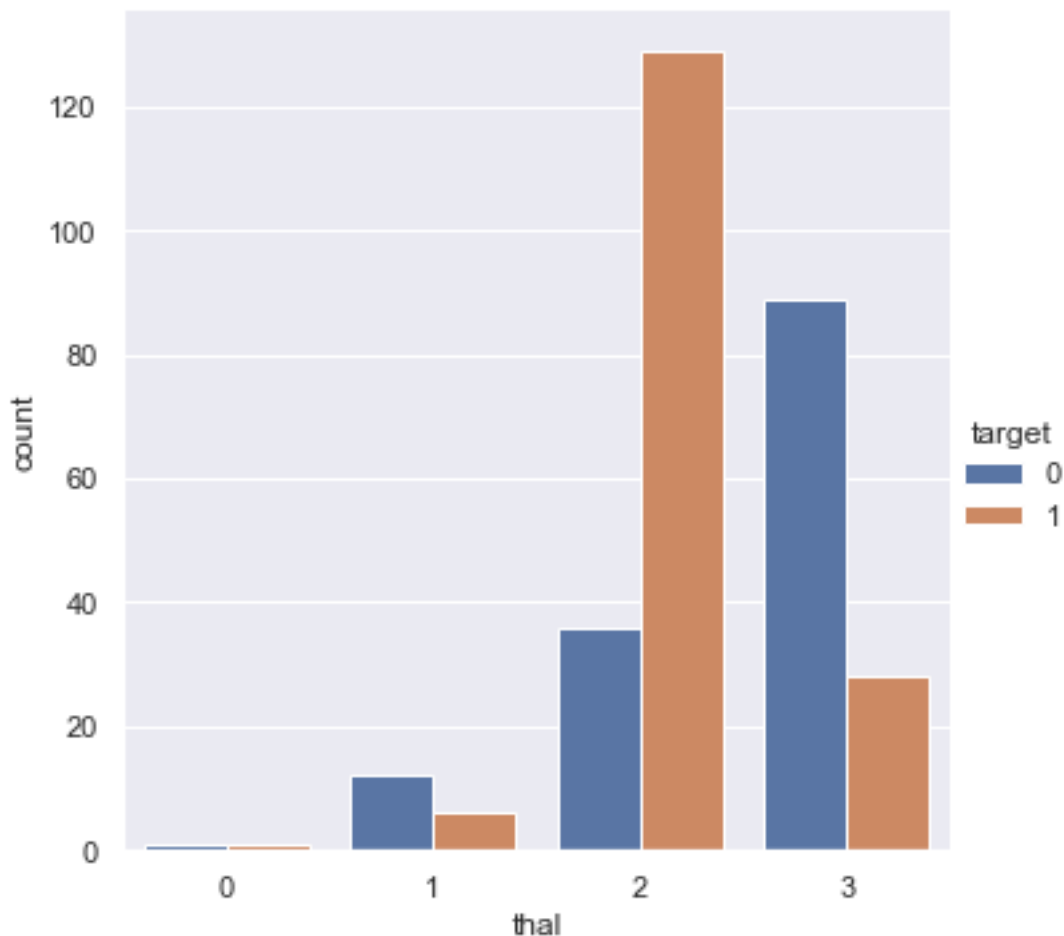
```
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\categorical.py:3704:  
UserWarning: The `factorplot` function has been renamed to `catplot`. The  
original name will be removed in a future release. Please update your code. Note  
that the default `kind` in `factorplot` (`'point'`) has changed to `strip` in  
`catplot`.
```

```
warnings.warn(msg)
```

```
C:\Users\nicme\anaconda3\lib\site-packages\seaborn\_decorators.py:36:  
FutureWarning: Pass the following variable as a keyword arg: x. From version  
0.12, the only valid positional argument will be `data`, and passing other  
arguments without an explicit keyword will result in an error or  
misinterpretation.
```

```
warnings.warn(
```

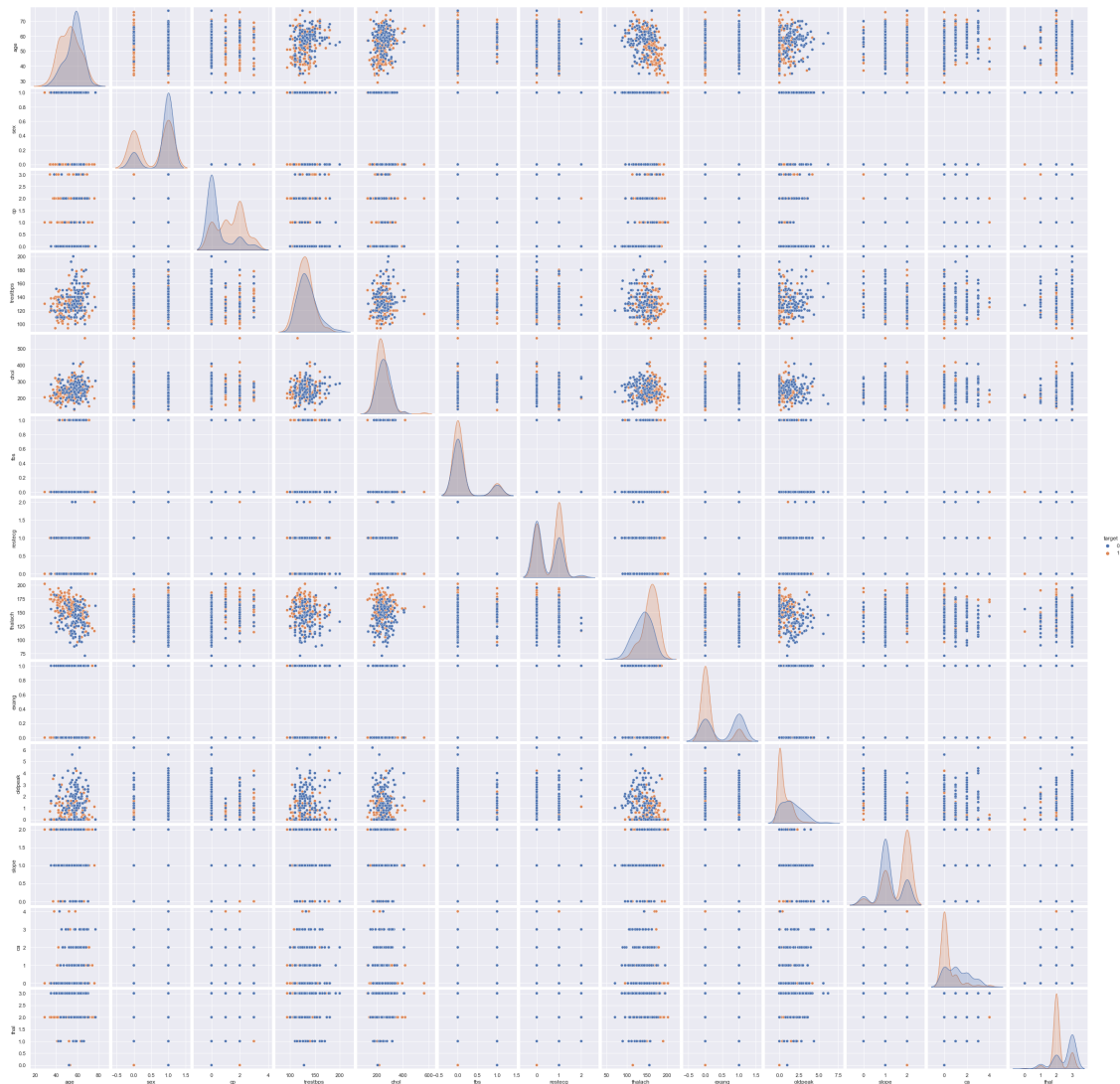
```
[53]: <seaborn.axisgrid.FacetGrid at 0x2136b980c70>
```



```
[54]: # People with a fixed defect have higher occurrences of CVD.
```

```
[55]: sns.pairplot(df , hue='target')
```

```
[55]: <seaborn.axisgrid.PairGrid at 0x2136d1102b0>
```



```
[65]: from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
```

```
[67]: df = df.dropna()
```

```
df = df.drop(columns = ['oldpeak', 'slope', 'ca', 'thal', 'fbs', 'restecg',
↳ 'exang'])
df = df.rename(columns = {'age': 'age', 'sex': 'gender', 'cp': 'chest pain',
↳ 'trestbps': 'blood pressure', 'chol': 'cholesterol level', 'thalach': 'max
↳ heart rate', })
df.head()
```

```
[67]:
```

	age	gender	chest pain	blood pressure	cholesterol level	max heart rate	\
0	63	1	3	145	233	150	
1	37	1	2	130	250	187	
2	41	0	1	130	204	172	
3	56	1	1	120	236	178	
4	57	0	0	120	354	163	

	target
0	1
1	1
2	1
3	1
4	1

```
[78]: df = shuffle(df)
x_train, x_test, y_train, y_test = train_test_split(df.iloc[:, :-1], df.iloc[:,
↳ -1], test_size=0.3)
```

```
[79]: from sklearn.linear_model import LogisticRegression
```

```
[80]: model = LogisticRegression()
```

```
[81]: model.fit(x_train, y_train)
```

```
[81]: LogisticRegression()
```

```
[82]: print(model.score(x_train, y_train))
print(model.score(x_test, y_test))
```

```
0.7819905213270142
0.7912087912087912
```

```
[83]: y_pred = model.predict(x_test)
print('Accuracy of logistic regression on test set:{:.2f}'.format(model.
↳ score(x_test, y_test)))
```

```
Accuracy of logistic regression on test set:0.79
```

```
[84]: from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test, y_pred)
print(confusion_matrix)
```



$$\begin{bmatrix} 31 & 12 \\ 7 & 41 \end{bmatrix}$$