Heart Disease Dataset Project

April 14, 2022

```
[74]: import pandas as pd
      import numpy as np
      import matplotlib.pyplot as plt
      import seaborn as sns
      import warnings
      warnings.filterwarnings('ignore')
```

```
[92]: df = pd.read_csv("heart.csv")
```

```
[93]: df.head()
```

```
[93]:    age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
      0   63    1   3       145   233    1        0      150      0      2.3      0
      1   37    1   2       130   250    0        1      187      0      3.5      0
      2   41    0   1       130   204    0        0      172      0      1.4      2
      3   56    1   1       120   236    0        1      178      0      0.8      2
      4   57    0   0       120   354    0        1      163      1      0.6      2

         ca  thal  target
      0   0     1       1
      1   0     2       1
      2   0     2       1
      3   0     2       1
      4   0     2       1
```

```
[94]: df.shape
```

```
[94]: (303, 14)
```

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

```
[96]:               age         sex          cp    trestbps        chol         fbs  \
      count  303.000000  303.000000  303.000000  303.000000  303.000000  303.000000
      mean    54.366337    0.683168    0.966997  131.623762  246.264026    0.148515
      std      9.082101    0.466011    1.032052   17.538143   51.830751    0.356198
      min     29.000000    0.000000    0.000000   94.000000  126.000000    0.000000
      25%     47.500000    0.000000    0.000000  120.000000  211.000000    0.000000
```

```
50%     55.000000    1.000000    1.000000  130.000000  240.000000    0.000000
75%     61.000000    1.000000    2.000000  140.000000  274.500000    0.000000
max     77.000000    1.000000    3.000000  200.000000  564.000000    1.000000

          restecg     thalach       exang     oldpeak       slope          ca  \
count  303.000000  303.000000  303.000000  303.000000  303.000000  303.000000
mean     0.528053  149.646865    0.326733    1.039604    1.399340    0.729373
std      0.525860   22.905161    0.469794    1.161075    0.616226    1.022606
min      0.000000   71.000000    0.000000    0.000000    0.000000    0.000000
25%      0.000000  133.500000    0.000000    0.000000    1.000000    0.000000
50%      1.000000  153.000000    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  303.000000  303.000000
mean     2.313531    0.544554
std      0.612277    0.498835
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
```

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

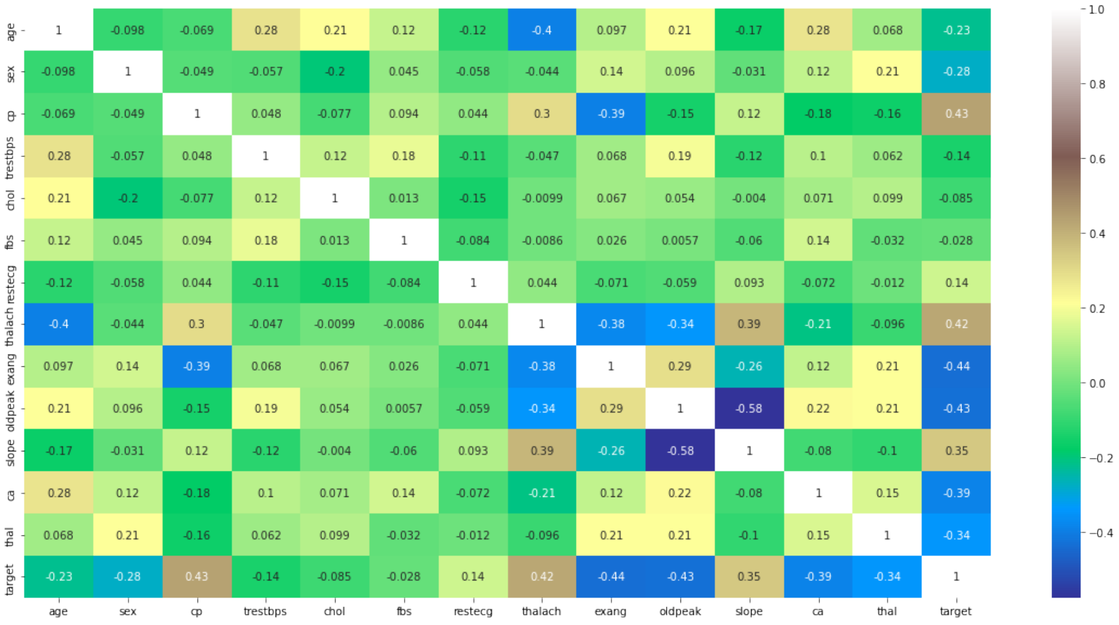
```
[97]: age         0
      sex         0
      cp          0
      trestbps    0
      chol        0
      fbs         0
      restecg     0
      thalach     0
      exang       0
      oldpeak     0
      slope       0
      ca          0
      thal        0
      target      0
      dtype: int64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

```
[98]: print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
None
```

[99]:
```
plt.figure(figsize=(20,10))
sns.heatmap(df.corr(), annot=True, cmap='terrain')
```

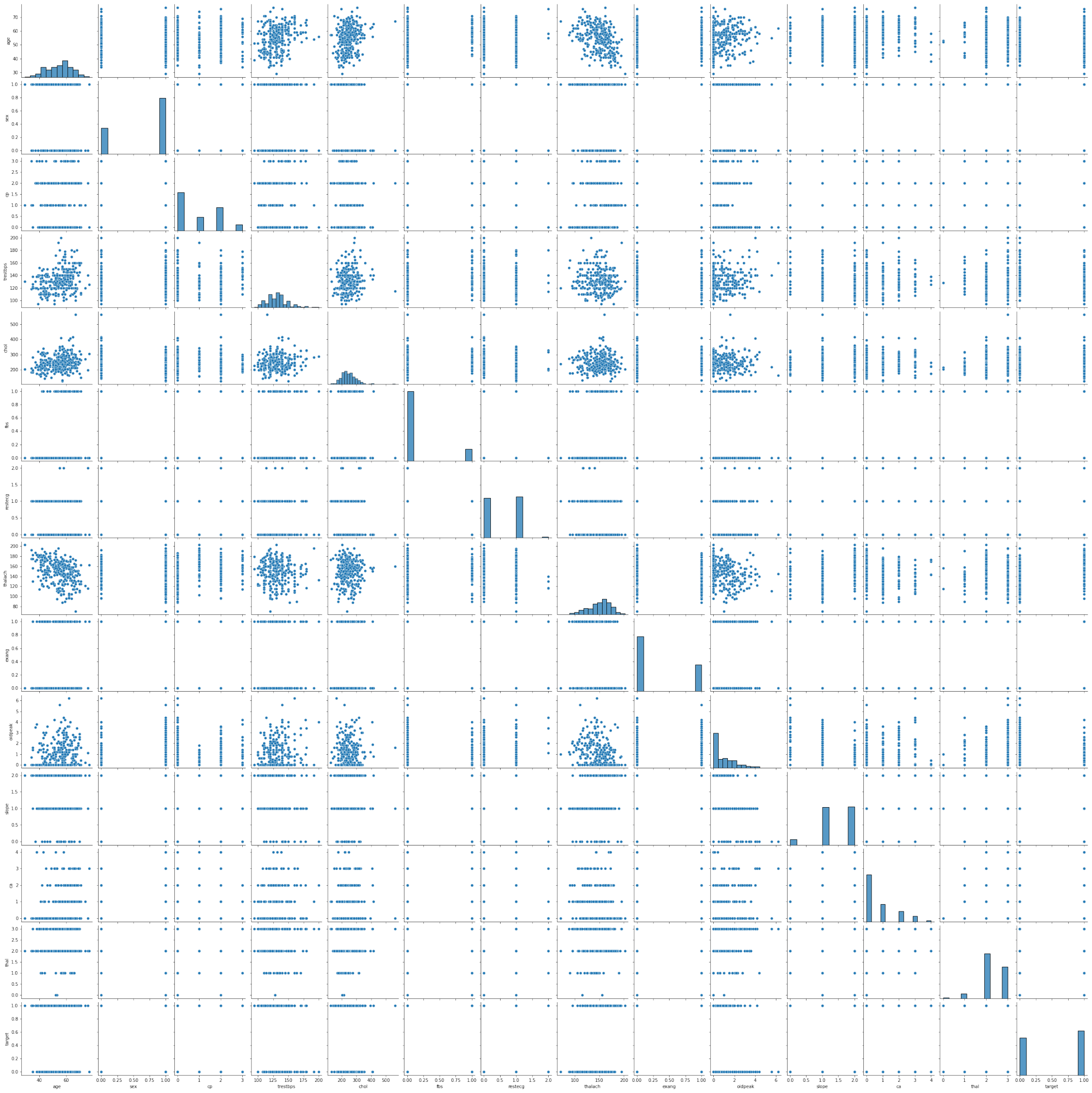
[99]:
```
<AxesSubplot:>
```



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1	-0.098	-0.069	0.28	0.21	0.12	-0.12	-0.4	0.097	0.21	-0.17	0.28	0.068	-0.23
sex	-0.098	1	-0.049	-0.057	-0.2	0.045	-0.058	-0.044	0.14	0.096	-0.031	0.12	0.21	-0.28
cp	-0.069	-0.049	1	0.048	-0.077	0.094	0.044	0.3	-0.39	-0.15	0.12	-0.18	-0.16	0.43
trestbps	0.28	-0.057	0.048	1	0.12	0.18	-0.11	-0.047	0.068	0.19	-0.12	0.1	0.062	-0.14
chol	0.21	-0.2	-0.077	0.12	1	0.013	-0.15	-0.0099	0.067	0.054	-0.004	0.071	0.099	-0.085
fbs	0.12	0.045	0.094	0.18	0.013	1	-0.084	-0.0086	0.026	0.0057	-0.06	0.14	-0.032	-0.028
restecg	-0.12	-0.058	0.044	-0.11	-0.15	-0.084	1	0.044	-0.071	-0.059	0.093	-0.072	-0.012	0.14
thalach	-0.4	-0.044	0.3	-0.047	-0.0099	-0.0086	0.044	1	-0.38	-0.34	0.39	-0.21	-0.096	0.42
exang	0.097	0.14	-0.39	0.068	0.067	0.026	-0.071	-0.38	1	0.29	-0.26	0.12	0.21	-0.44
oldpeak	0.21	0.096	-0.15	0.19	0.054	0.0057	-0.059	-0.34	0.29	1	-0.58	0.22	0.21	-0.43
slope	-0.17	-0.031	0.12	-0.12	-0.004	-0.06	0.093	0.39	-0.26	-0.58	1	-0.08	-0.1	0.35
ca	0.28	0.12	-0.18	0.1	0.071	0.14	-0.072	-0.21	0.12	0.22	-0.08	1	0.15	-0.39
thal	0.068	0.21	-0.16	0.062	0.099	-0.032	-0.012	-0.096	0.21	0.21	-0.1	0.15	1	-0.34
target	-0.23	-0.28	0.43	-0.14	-0.085	-0.028	0.14	0.42	-0.44	-0.43	0.35	-0.39	-0.34	1

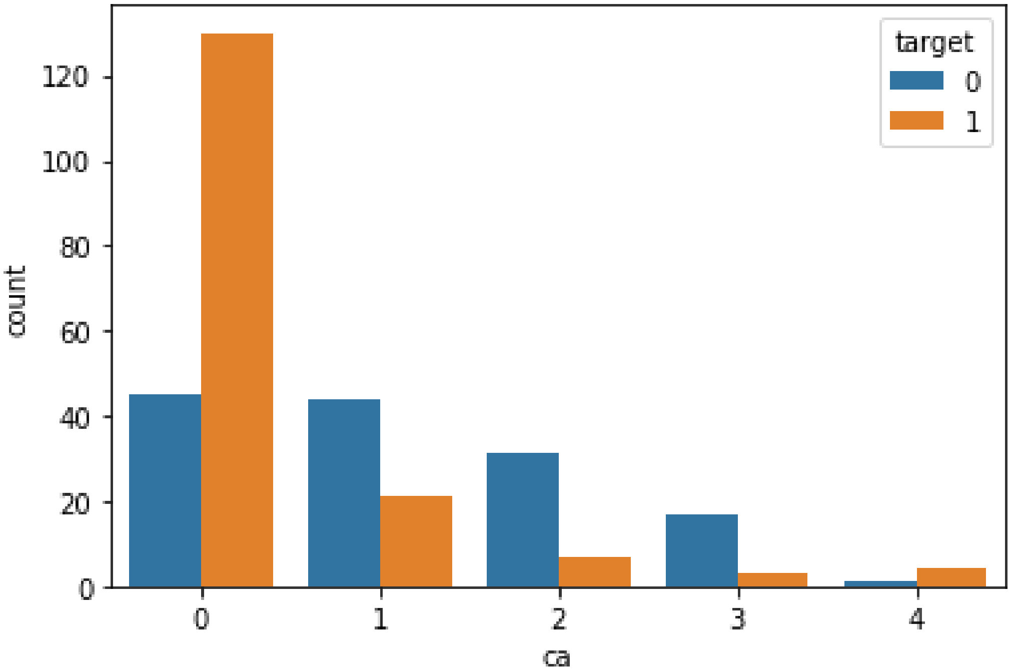
[101]:
```
sns.pairplot(data=df)
```

```
[101]: <seaborn.axisgrid.PairGrid at 0x1f9501cb7f0>
```



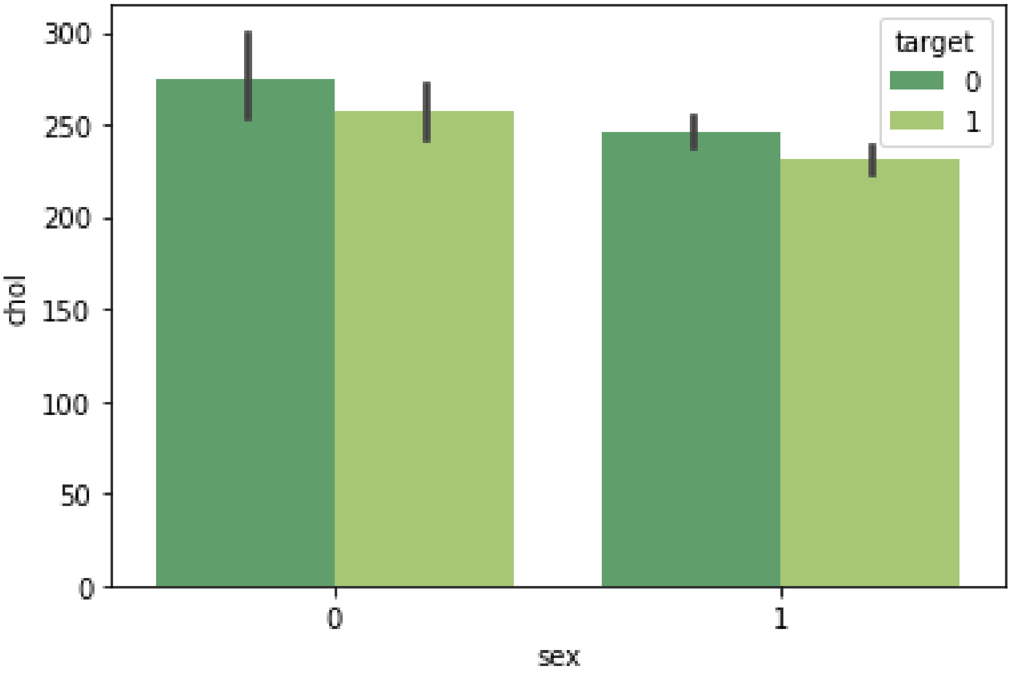
```
[102]: sns.countplot(x='ca',hue='target',data=df)
```

```
[102]: <AxesSubplot:xlabel='ca', ylabel='count'>
```



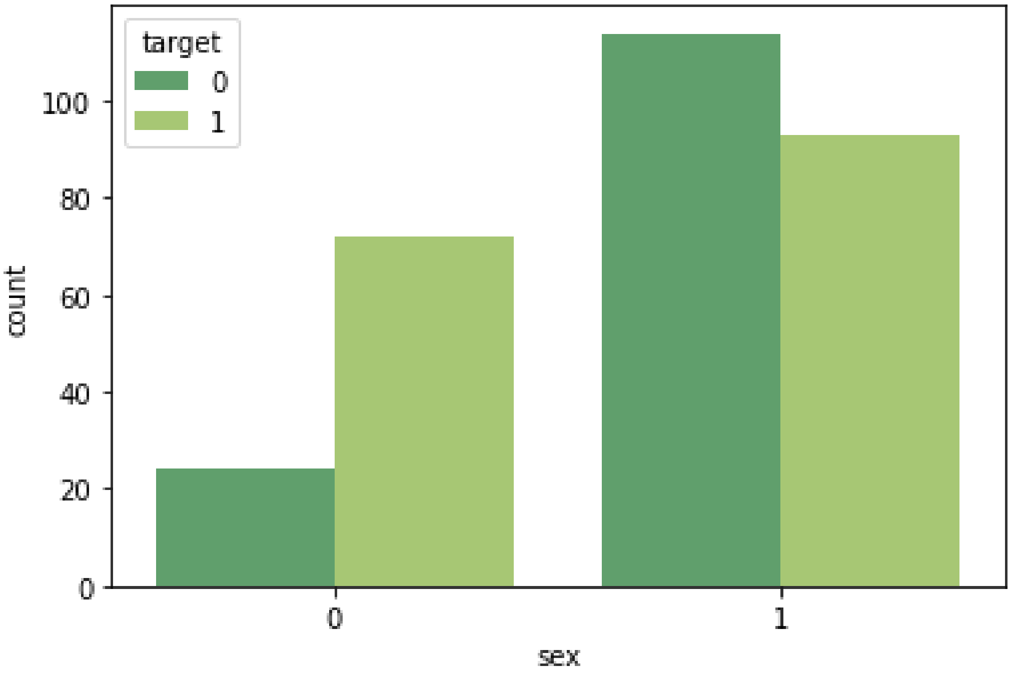
[105]:
```
sns.barplot(data=df, x='sex', y='chol', hue='target', palette='summer')
```

[105]:
```
<AxesSubplot:xlabel='sex', ylabel='chol'>
```



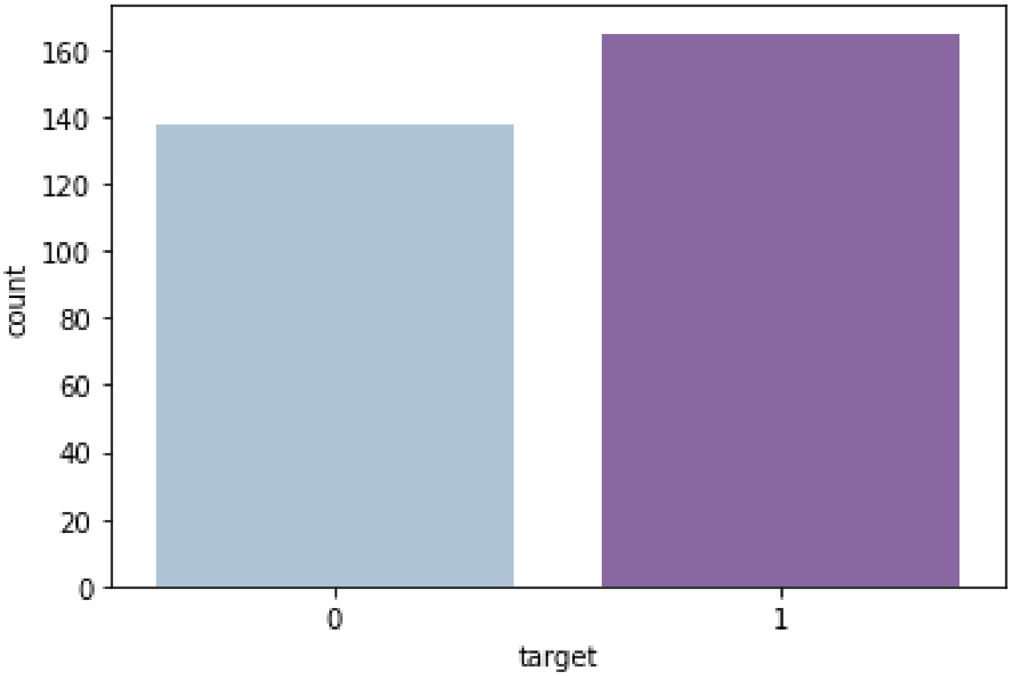
[113]:
```
sns.countplot(x='sex', data=df, palette='summer', hue='target')
```

[113]:
```
<AxesSubplot:xlabel='sex', ylabel='count'>
```



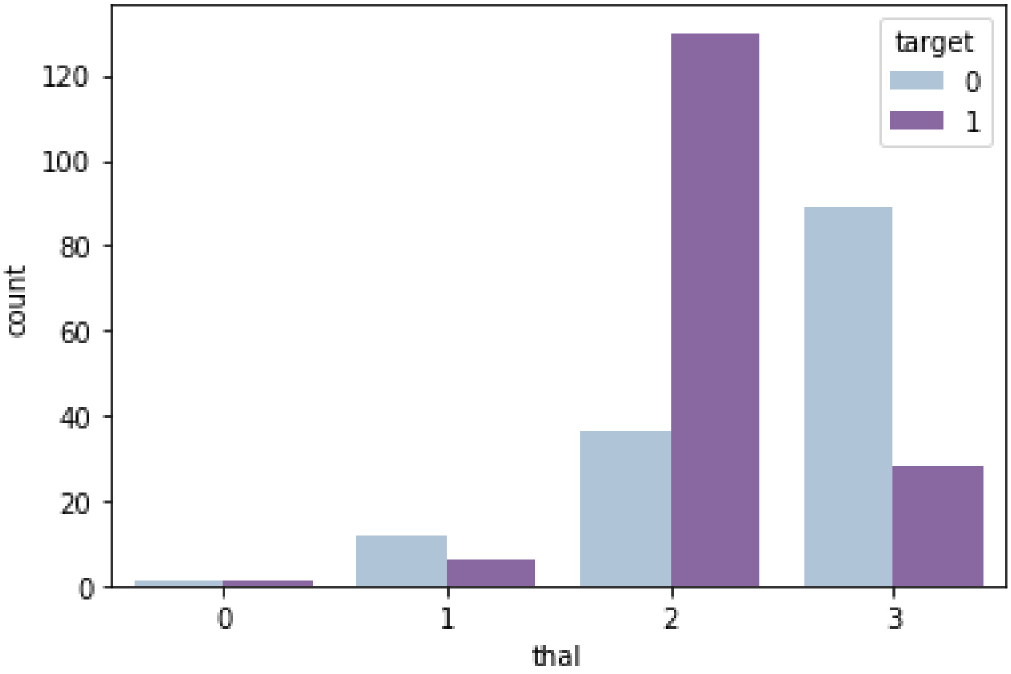
[119]:
```
sns.countplot(x='target',palette='BuPu', data=df)
```

[119]:
```
<AxesSubplot:xlabel='target', ylabel='count'>
```



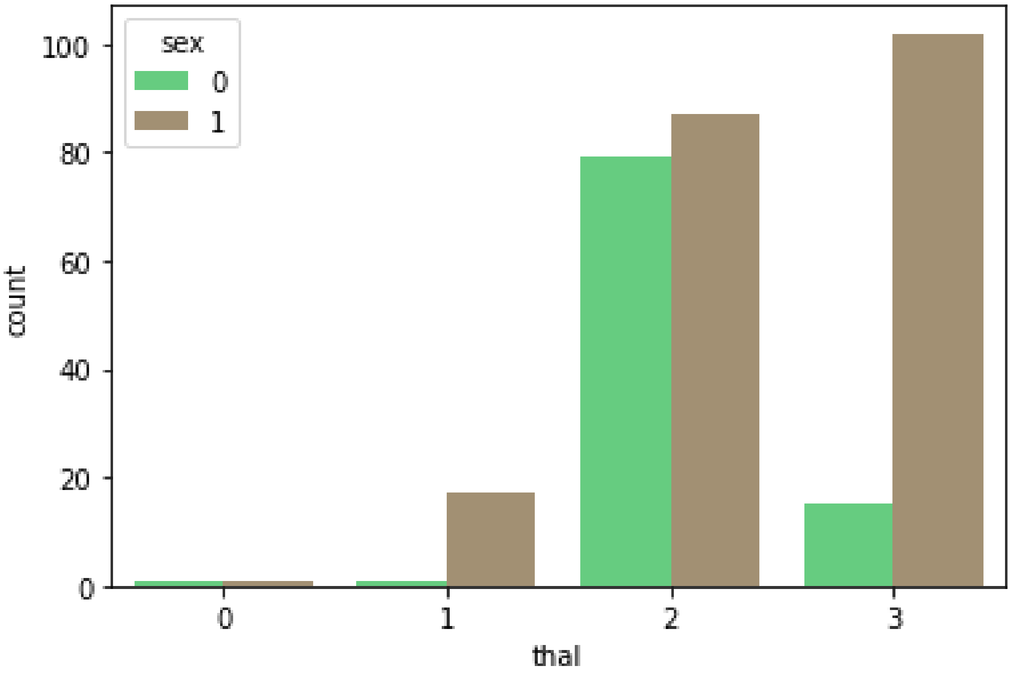
```
[108]: sns.countplot(x='thal',data=df, hue='target', palette='BuPu' )
```

```
[108]: <AxesSubplot:xlabel='thal', ylabel='count'>
```



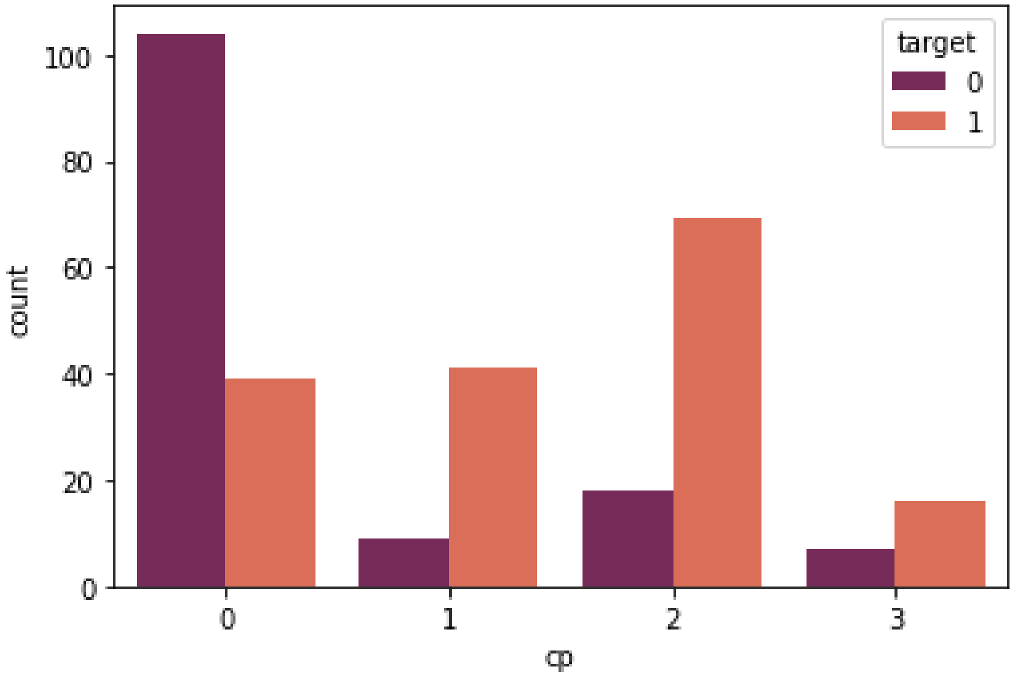
```
[109]: sns.countplot(x='thal', hue='sex',data=df, palette='terrain')
```

```
[109]: <AxesSubplot:xlabel='thal', ylabel='count'>
```



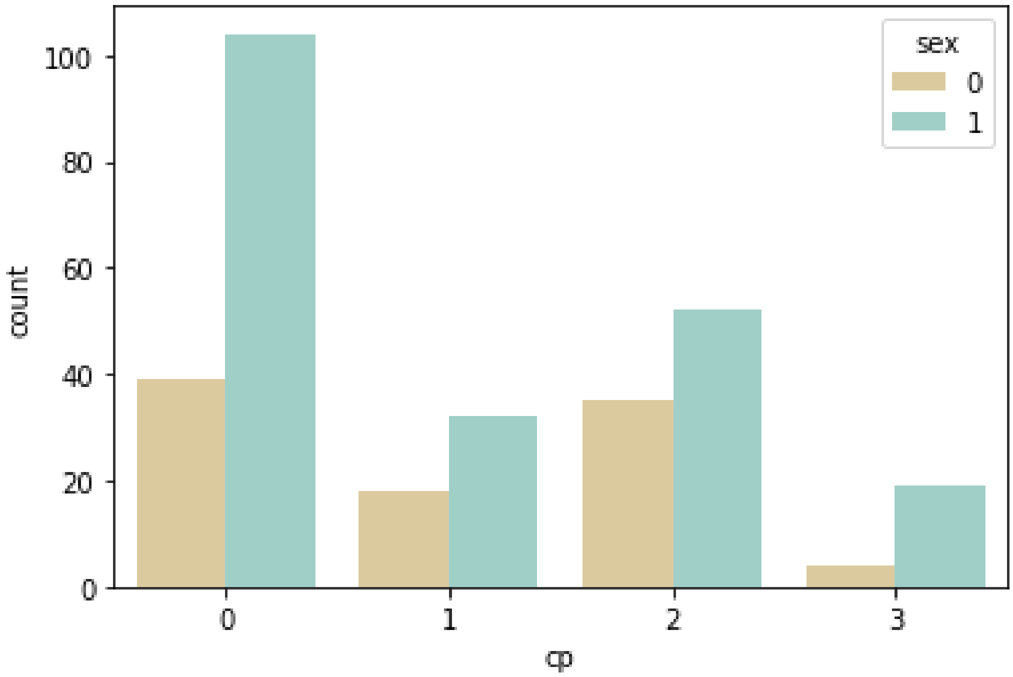
```
[110]: sns.countplot(x='cp' ,hue='target', data=df, palette='rocket')
```

```
[110]: <AxesSubplot:xlabel='cp', ylabel='count'>
```



```
[111]: sns.countplot(x='cp', hue='sex',data=df, palette='BrBG')
```

```
[111]: <AxesSubplot:xlabel='cp', ylabel='count'>
```



[120]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
StandardScaler = StandardScaler()
columns_to_scale = ['age','trestbps','chol','thalach','oldpeak']
df[columns_to_scale] = StandardScaler.fit_transform(df[columns_to_scale])
```

[121]:
```
df.head()
```

[121]:
```
        age  sex  cp  trestbps      chol  fbs  restecg   thalach  exang  \
0  0.952197    1   3  0.763956 -0.256334    1        0  0.015443      0
1 -1.915313    1   2 -0.092738  0.072199    0        1  1.633471      0
2 -1.474158    0   1 -0.092738 -0.816773    0        0  0.977514      0
3  0.180175    1   1 -0.663867 -0.198357    0        1  1.239897      0
4  0.290464    0   0 -0.663867  2.082050    0        1  0.583939      1

    oldpeak  slope  ca  thal  target
0  1.087338      0   0     1       1
1  2.122573      0   0     2       1
2  0.310912      2   0     2       1
3 -0.206705      2   0     2       1
4 -0.379244      2   0     2       1
```

[122]:
```
X= df.drop(['target'], axis=1)
y= df['target']
```

[123]:
```
X_train, X_test,y_train, y_test=train_test_split(X,y,test_size=0.
 ↪3,random_state=40)
```

[124]:
```
print('X_train-', X_train.size)
print('X_test-',X_test.size)
print('y_train-', y_train.size)
print('y_test-', y_test.size)
```

```
X_train- 2756
X_test- 1183
y_train- 212
y_test- 91
```

Model Logistic Regression

[126]:
```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()

model1=lr.fit(X_train,y_train)
prediction1=model1.predict(X_test)
```

[127]:

```
from sklearn.metrics import confusion_matrix

cm=confusion_matrix(y_test,prediction1)
cm
```

[127]:

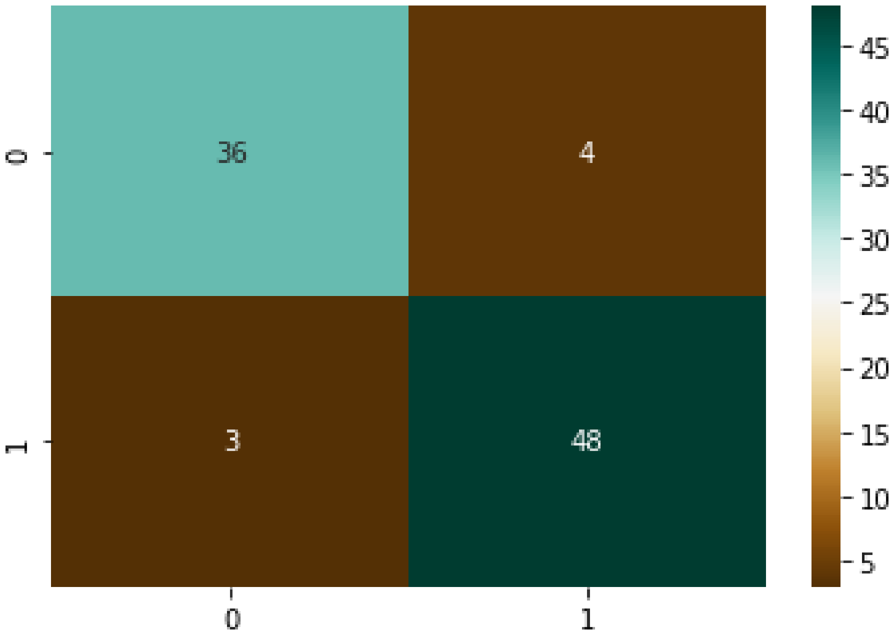
```
array([[36,  4],
       [ 3, 48]], dtype=int64)
```

[129]:

```
sns.heatmap(cm, annot=True,cmap='BrBG')
```

[129]:

```
<AxesSubplot:>
```



[130]:

```
TP=cm[0][0]
TN=cm[1][1]
FN=cm[1][0]
FP=cm[0][1]
print('Testing Accuracy:',(TP+TN)/(TP+TN+FN+FP))
```

```
Testing Accuracy: 0.9230769230769231
```

[131]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,prediction1)
```

[131]:

```
0.9230769230769231
```

```
[132]: from sklearn.metrics import classification_report
       print(classification_report(y_test, prediction1))
```

```
              precision    recall  f1-score   support

           0       0.92      0.90      0.91        40
           1       0.92      0.94      0.93        51

    accuracy                           0.92        91
   macro avg       0.92      0.92      0.92        91
weighted avg       0.92      0.92      0.92        91
```

DECISION TREE

```
[134]: from sklearn.tree import DecisionTreeClassifier

       dtc=DecisionTreeClassifier()
       model2=dtc.fit(X_train,y_train)
       prediction2=model2.predict(X_test)
       cm2= confusion_matrix(y_test,prediction2)
```

```
[135]: cm2
```

```
[135]: array([[33,  7],
              [15, 36]], dtype=int64)
```

```
[136]: accuracy_score(y_test,prediction2)
```

```
[136]: 0.7582417582417582
```

```
[137]: print(classification_report(y_test, prediction2))
```

```
              precision    recall  f1-score   support

           0       0.69      0.82      0.75        40
           1       0.84      0.71      0.77        51

    accuracy                           0.76        91
   macro avg       0.76      0.77      0.76        91
weighted avg       0.77      0.76      0.76        91
```

```
[139]: from sklearn.model_selection import KFold
       from sklearn.model_selection import cross_val_score
```

```
[140]: from sklearn.svm import SVC
```

```
svm=SVC()
model4=svm.fit(X_train,y_train)
prediction4=model4.predict(X_test)
cm4= confusion_matrix(y_test,prediction4)
```

[141]:
```
cm4
```

[141]:
```
array([[33,  7],
       [ 2, 49]], dtype=int64)
```

[142]:
```
accuracy_score(y_test, prediction4)
```

[142]:
```
0.9010989010989011
```

[143]:
```
from sklearn.naive_bayes import GaussianNB

NB = GaussianNB()
model5 = NB.fit(X_train, y_train)
prediction5 = model5.predict(X_test)
cm5= confusion_matrix(y_test, prediction5)
```

[144]:
```
cm5
```

[144]:
```
array([[35,  5],
       [ 6, 45]], dtype=int64)
```

[145]:
```
accuracy_score(y_test, prediction5)
```

[145]:
```
0.8791208791208791
```

[146]:
```
print('cm4', cm4)
print('-----------')
print('cm5',cm5)
```

```
cm4 [[33  7]
 [ 2 49]]
-----------
cm5 [[35  5]
 [ 6 45]]
```

[147]:
```
from sklearn.neighbors import KNeighborsClassifier

KNN = KNeighborsClassifier()
model6 = KNN.fit(X_train, y_train)
prediction6 = model6.predict(X_test)
cm6= confusion_matrix(y_test, prediction5)
cm6
```

```
[147]: array([[35,  5],
              [ 6, 45]], dtype=int64)
```

```
[149]: print('KNN :', accuracy_score(y_test, prediction6))
       print('LR :', accuracy_score(y_test, prediction1))
       print('DT :', accuracy_score(y_test, prediction2))
       print('NB: ', accuracy_score(y_test, prediction4))
       print('SVM :', accuracy_score(y_test, prediction5))
```

```
KNN : 0.8351648351648352
LR : 0.9230769230769231
DT : 0.7582417582417582
NB:  0.9010989010989011
SVM : 0.8791208791208791
```

Best Accuracy: Logistic Regression : 92 Same Accurancy NB and Decision tree : 90