In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
df = pd.read_csv('heart.csv')
```

In [3]:

```
df.head()
```

Out[3]:

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

In [4]:

```
df.shape
```

Out[4]:

```
(303, 14)
```

In [5]:

```
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
```

In [6]:

```
from sklearn import preprocessing
import matplotlib.pyplot as plt
plt.rc("font", size=14)
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
import seaborn as sns
sns.set(style="white")
sns.set(style="whitegrid", color_codes=True)
```

DATA EXPLORATION

In [7]:

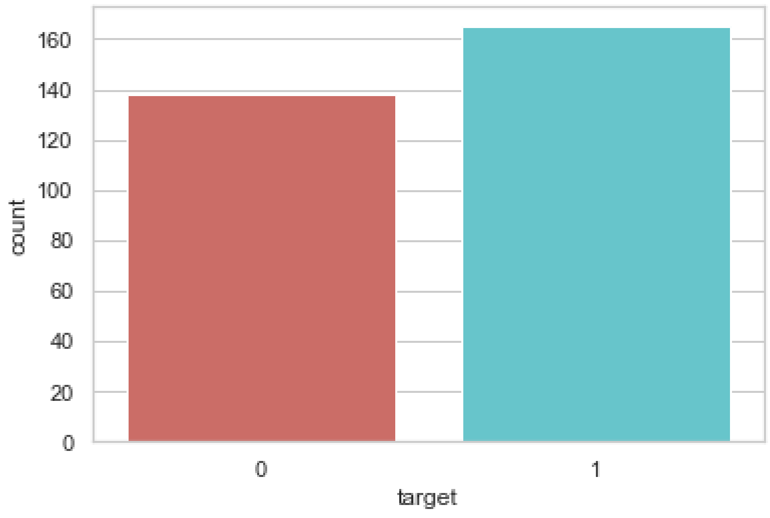
```
df['target'].value_counts()
```

Out[7]:

```
1    165
0    138
Name: target, dtype: int64
```

In [8]:

```
sns.countplot(x = 'target',data = df, palette = 'hls')
plt.show()
plt.savefig('count_plot')
```



```
<Figure size 432x288 with 0 Axes>
```

In [9]:

```
count_no_sub = len(df[df['target']==0])
count_sub = len(df[df['target']==1])
pct_of_no_sub = count_no_sub/(count_no_sub+count_sub)
print("percentage of no is", pct_of_no_sub*100)
pct_of_sub = count_sub/(count_no_sub+count_sub)
print("percentage of yes is", pct_of_sub*100)
```

```
percentage of no is 45.54455445544555
percentage of yes is 54.45544554455446
```

In [10]:

```
df.groupby('target').mean()
```

Out[10]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach
target								
0	56.601449	0.826087	0.478261	134.398551	251.086957	0.159420	0.449275	139.101449
1	52.496970	0.563636	1.375758	129.303030	242.230303	0.139394	0.593939	158.466667

In [14]:

```
X = df.drop(columns = 'target', axis =1)
Y = df['target']
```

In [15]:

```
X.head()
```

Out[15]:

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

In [16]:

```
Y.head()
```

Out[16]:

```
0    1
1    1
2    1
3    1
4    1
Name: target, dtype: int64
```

In [25]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.4, stratify = Y, ran
```

In [26]:

```
print(X.shape,X_train.shape,X_test.shape)
```

```
(303, 13) (181, 13) (122, 13)
```

In [27]:

```
model = LogisticRegression()
```

In [28]:

```
model.fit(X_train,Y_train)
```

```
C:\Users\satya\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.p
y:763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scik
it-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regre
ssion (https://scikit-learn.org/stable/modules/linear_model.html#logistic-re
gression)
  n_iter_i = _check_optimize_result(
```

Out[28]:

```
LogisticRegression()
```

In [29]:

```
from sklearn.metrics import accuracy_score
```

In [30]:

```
X_train_prediction = model.predict(X_train)
train_data_accuracy = accuracy_score(X_train_prediction, Y_train)
```

In [31]:

```
train_data_accuracy
```

Out[31]:

```
0.8784530386740331
```

In [32]:

```
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)
```

In [33]:

```
test_data_accuracy
```

Out[33]:

```
0.8360655737704918
```

In [44]:

```
input_data = (56,0,0,134,409,0,0,150,1,1.9,1,2,3)
input_data_as_numpy_array = np.asarray(input_data)
```

In [45]:

```
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
prediction = model.predict(input_data_reshaped)
```

In [46]:

```
prediction
```

Out[46]:

```
array([0], dtype=int64)
```

In [47]:

```
if (prediction[0] ==0 ):
    print('The person does not have a heart problem')
else:
    print('The person does have a heart problem')
```

```
The person does not have a heart problem
```

In [ ]: