Experiment 3

Diagnose disease risk from Patient data.

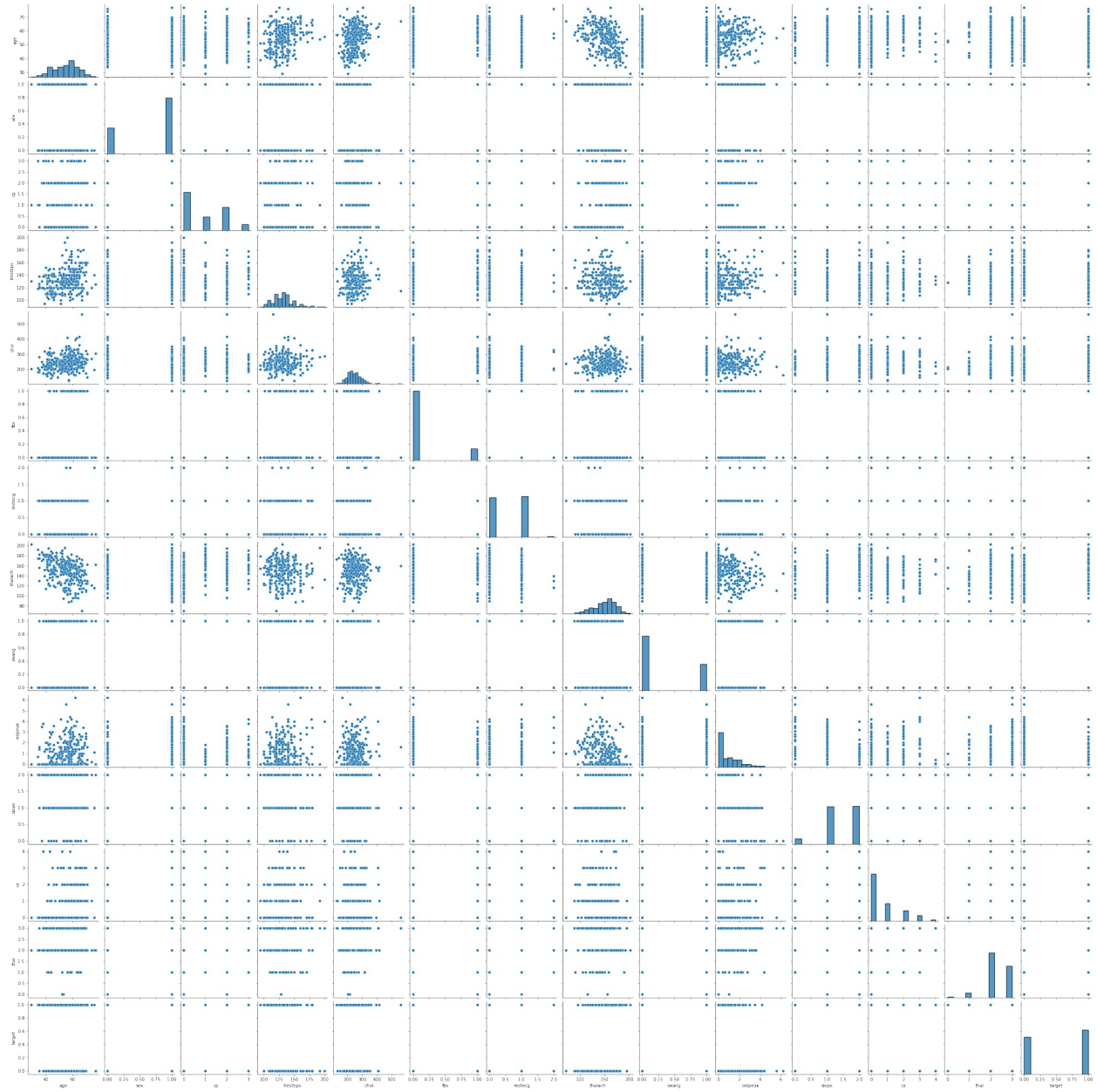
Reference: https://youtu.be/DkdHmc1r4gk?feature=shared

```
import numpy as np
import pandas as pd

df=pd.read_excel("1645792390_cep1_dataset.xlsx")

df.info()
```

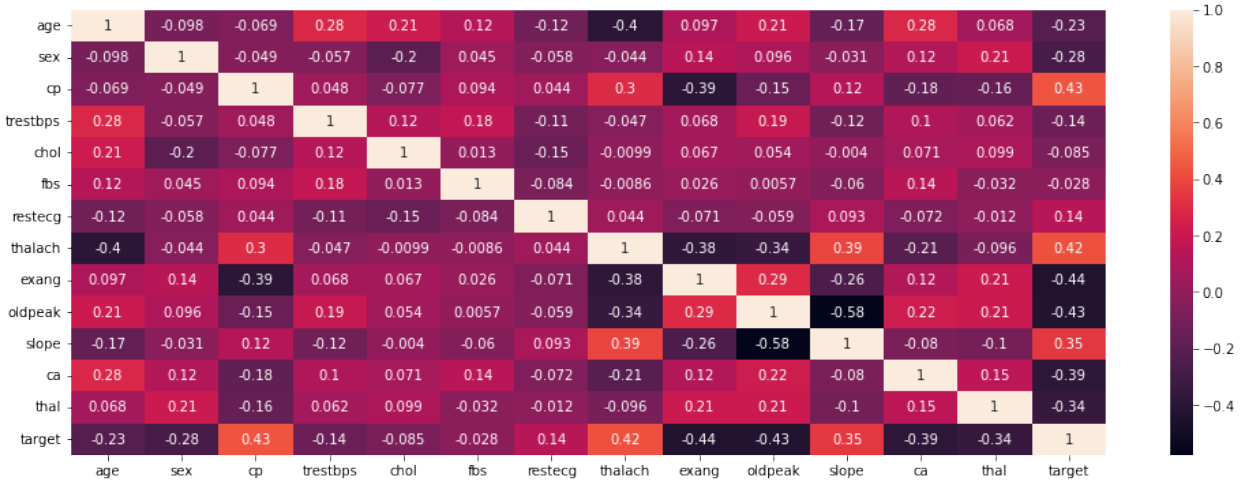
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 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
```

```
import seaborn as sns

sns.pairplot(df)
```

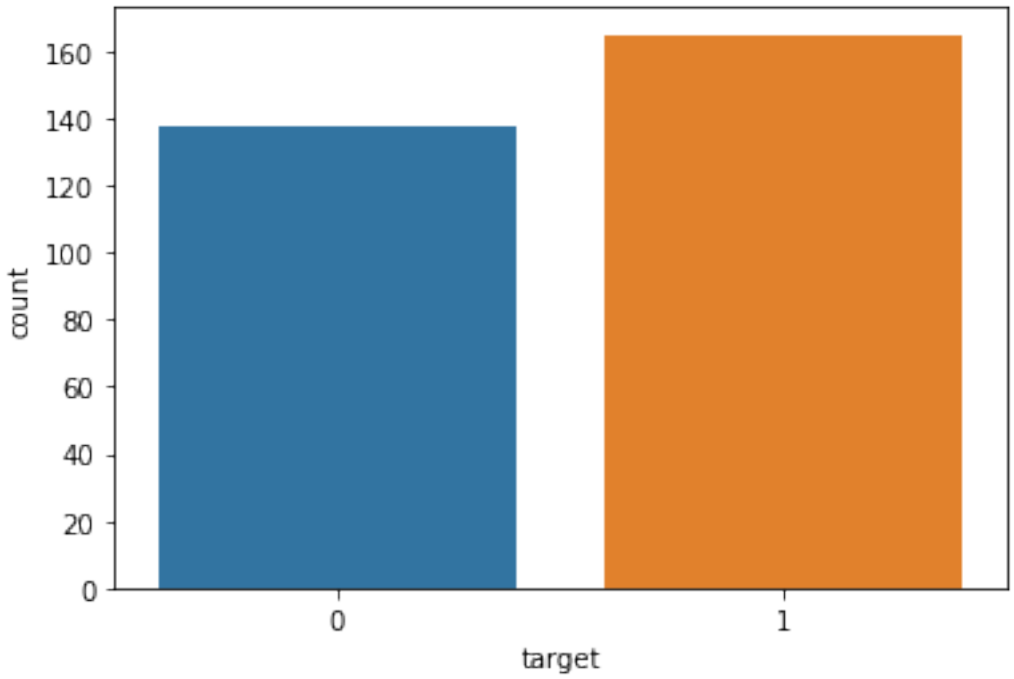
```
<seaborn.axisgrid.PairGrid at 0x1afb061bf40>
```



```
import matplotlib.pyplot as plt
plt.figure(figsize=(17,6))
sns.heatmap(df.corr(),annot=True)
plt.show()
```




```
sns.countplot(x="target", data=df)
plt.show()
```




```
X=df.drop('target', axis=1) ## independent Variables

Y=df["target"] ## dependent Variable

X
```

```
     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
0        0   0     1
1        0   0     2
2        2   0     2
3        2   0     2
4        2   0     2
..     ...  ..   ...
298      1   0     3
299      1   0     3
300      1   2     3
301      1   1     3
302      1   1     2

[303 rows x 13 columns]

Y

0      1
1      1
2      1
3      1
4      1
      ..
298    0
299    0
300    0
301    0
302    0
Name: target, Length: 303, dtype: int64
```

Using Logistic regression model to predict disease

```
##Logistic Regression Model
from sklearn import datasets,linear_model
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=0.3, random_state=42)

logmodel=LogisticRegression()

logmodel.fit(X_train, y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\
_logistic.py:814: 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
Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

```
LogisticRegression()

predictions=logmodel.predict(X_test)

from sklearn.metrics import classification_report

classification_report(y_test,predictions)
```

```
'              precision    recall  f1-score   support\n\n           0
0.80      0.78      0.79        41\n           1       0.82      0.84
0.83        50\n\n    accuracy                           0.81
91\n   macro avg       0.81      0.81      0.81        91\nweighted
avg       0.81      0.81      0.81        91\n'

from sklearn.metrics import accuracy_score

accuracy_score(y_test,predictions)

0.8131868131868132
```

Experiment 4

The primary purpose of visualization is to help medical staff interpret data analytics results faster, recognize trends, and make better decisions

How Many People Have Heart Disease, And How Many Don't Have Heart Disease In This Dataset?

```
target : 0=less chance of heart attack
         1=more chance of heart attack

df.columns
```

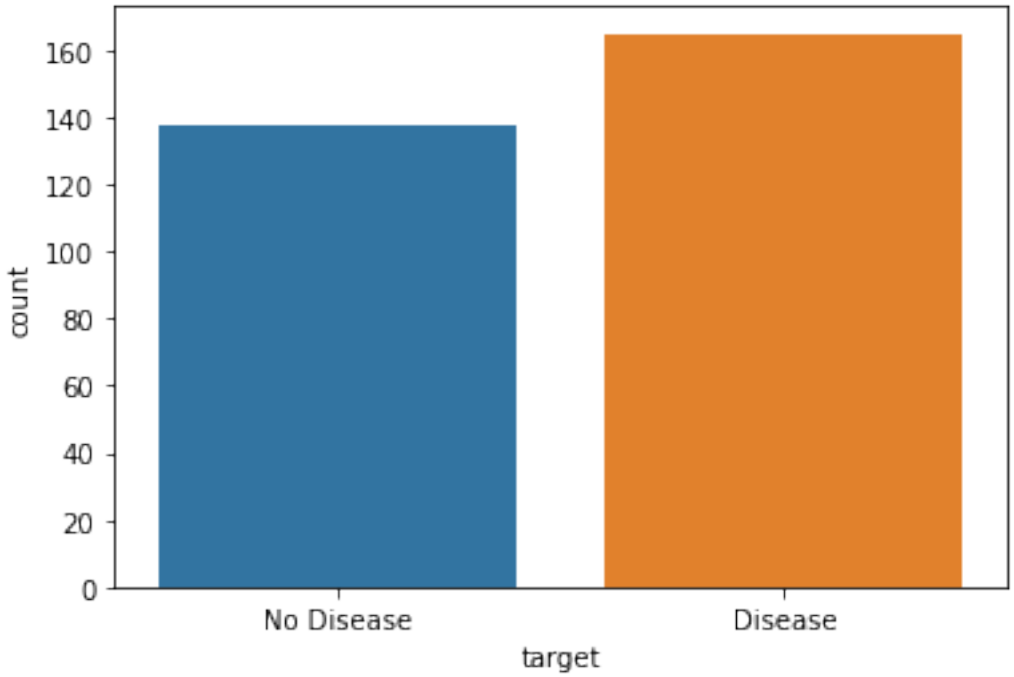
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

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

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

```
sns.countplot(df['target'])
plt.xticks([0,1],['No Disease','Disease'])
plt.show()
```

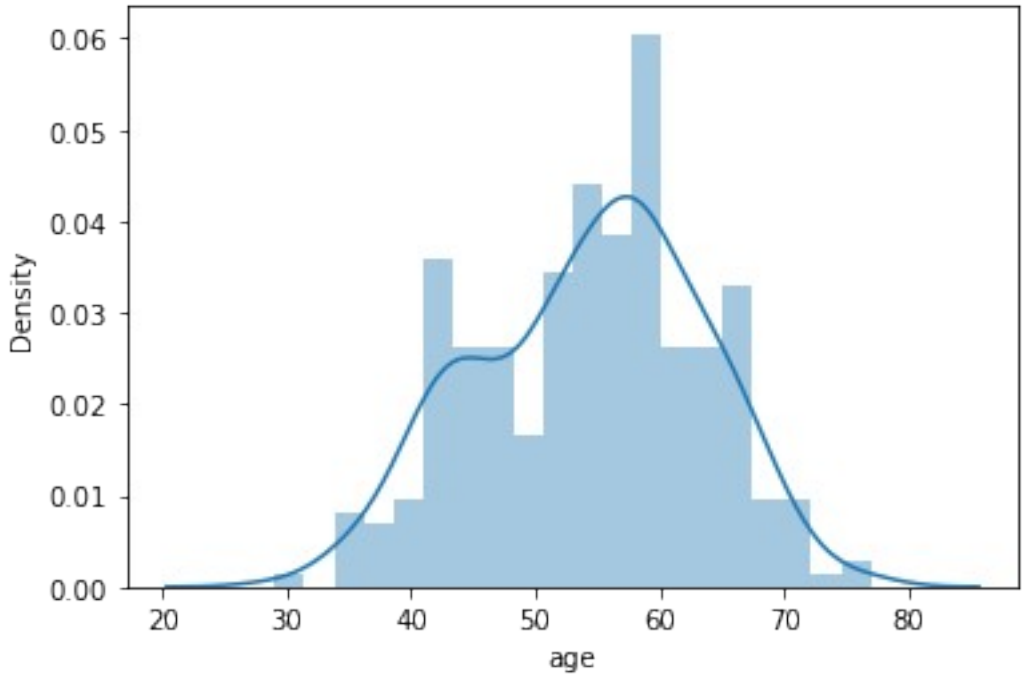
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  warnings.warn(
```



which age patient have hign chances of heart disease?

```
sns.distplot(df['age'],bins=20)
plt.show()
```

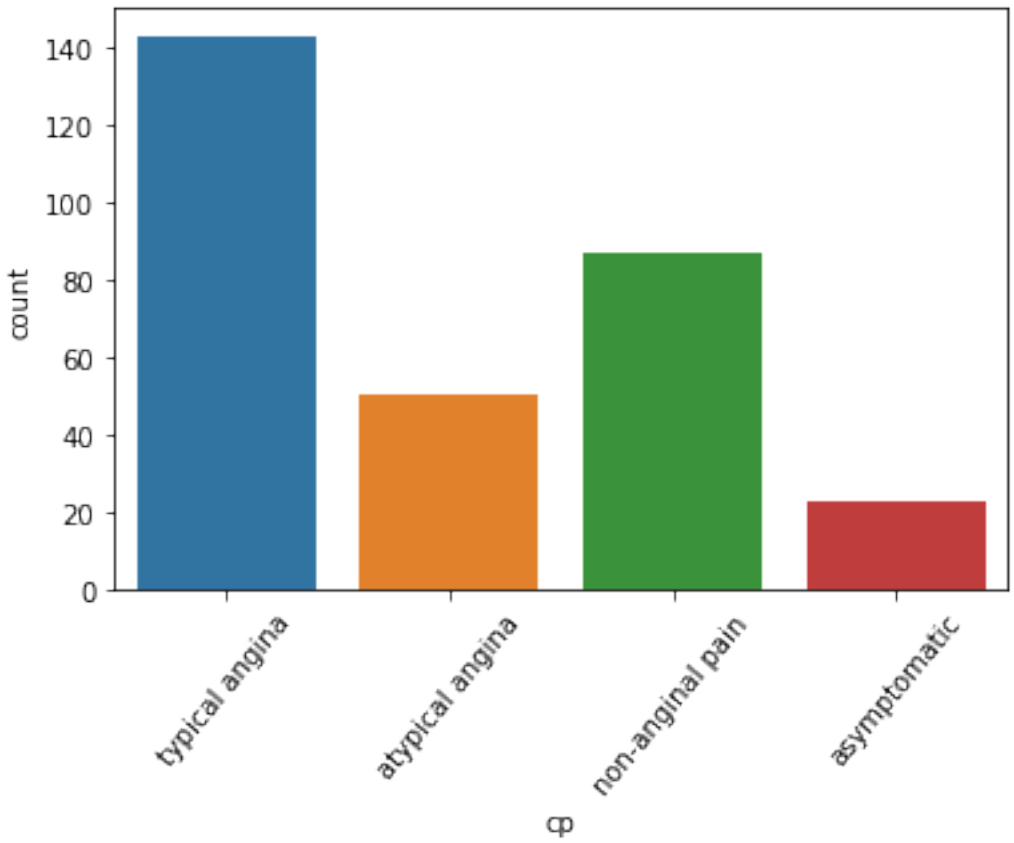
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\
distributions.py:2619: FutureWarning: `distplot` is a deprecated
function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Which chest pain type is more common between people?

chest pain types:

```
       0: typical angina
       1: atypical angina
       3: non-anginal pain
       4: asymptomatic

sns.countplot(df['cp'])
plt.xticks([0,1,2,3],["typical angina","atypical angina","non-anginal
pain","asymptomatic"])
plt.xticks(rotation=50)
plt.show()

C:\ProgramData\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(
```



chest pain type "typical angina" is more common between people.

which are categorical and non-categorical columns?

```
df.columns
```

```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg',
'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

```
cate_val=[] #categorical columns
cont_val=[] # non-categorical columns

for column in df.columns:
    if df[column].nunique() <=10:
        cate_val.append(column)
    else:
        cont_val.append(column)

cate_val
```

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

```
cont_val
```

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

```
df.hist(cont_val, figsize=(20,18))
plt.tight_layout()
plt.show()
```