```
# -*- coding: utf-8 -*-
"""
Created on Mon Jul 26 09:57:16 2021

@author: oseia
"""

# IMPORTING THE DATA : #

import pandas as pd
import numpy as np
cv = pd.read_excel("data.xlsx")

pd.set_option('display.max_columns',None)
pd.set_option('display.max_rows',None)



# PRELIMINARY DATA INPECTION : #

cv.info()
cv.isna().sum()
cv[cv.duplicated()]




# #FINDINGS AND DATA UNDERSTANDING # #

       # This data consists of 14 columns and 303 rows and has a dataframe
structure. All the variables(columns) seem to be numeric type(but in
actual sense some are supposed to be categorical)
       # of which 13 are integers and the other is a float type.All these
variables are factors
       # that can have an impact on heart heath,though, we are yet to
acertain if all have the potential of causing heart problem or some of
them.


       # There are no missing values in this data but has a duplicated
row(row 163 and index 164) and one of them needs to be dropped as soon as
possible.The numeric values and variables are not in comparable
       # ranges and need to be normalized or standardized..

       # We want to identify the factors that have impact on heart
health,develop a model using logistic regression
       # to predict and know the individuals that are more likely to be
prone to heart attack.

       #In furtherance, the target variable here is target and all the
other variables(13) like age,sex,ca,cp,trestbps,fbs,thal,thalach,etc will
be our independent variables..
```

```
#REMOVING THE DUPLICATES FROM THE DATA : #

cv = cv[~(cv.duplicated(keep='first'))]
cv.info() #NB: After removing the duplicate,our data now has 303 rows..





#STATISTICAL SUMMARY OF THE DATA AND OUTLIER DETECTION:

cv.describe()


import matplotlib.pyplot as plt
import seaborn as sns





        #AGE (Continuous and Unimodal Distribution,it follows normal
distribution.
        #            It has a wider range and higher variance.This means
it is rich in information.)

cv.age.describe()

cv.age.skew()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.age, kde = True, bins = 5)
plt.show()


                  #Outlier Checking Using Boxplot (No outliers)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.age, color = 'mediumaquamarine',)
plt.show()


#                         #Normality Test
# from scipy.stats import shapiro
# shapiro(cv.age)
```

```
       # SEX  (Categorical- Bimodal distribution)

cv.describe()
cv.sex.describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv. sex , kde = True, bins = 5,)
plt.show()



               #Outlier Checking Using Boxplot  (No outliers)

                     plt.figure(figsize= (15, 5))
                     sns.boxplot(cv.sex, color = 'mediumaquamarine',)
                     plt.show()




       # CP (Categorical variable- Multimodal Distribution )

cv.describe()
cv.cp.describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv. cp , kde = True, bins = 5,)
plt.show()



              #Outlier Checking Using Boxplot  (No outliers)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.cp, color = 'mediumaquamarine',)
plt.show()




       ## TRESTBPS  (Continuous variable- Unimodal Distribution ,follows
normal distribution)

cv.describe()
cv.trestbps .describe()

cv.trestbps.skew()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.trestbps, kde = True, bins = 5,)
plt.show()
```

```
            #Outlier Checking Using Boxplot  (There seems to be some
outliers in this variable but they are all points of influence and
therefore no need to treat them)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.trestbps, color = 'mediumaquamarine',)
plt.show()


q1=120  ; q3=140
q1 - 1.5* (q3 - q1)
q3 + 1.5* (q3 - q1)
cv[(cv.trestbps > 170) | ( cv.trestbps < 90)]['trestbps'] #NB :
Medically,trestbps above 129 is high






     ## CHOL (Continuous variable- Unimodal Distribution ,it follows
normal distribution)

cv.describe()
cv.chol .describe()

cv.chol.skew()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.chol, kde = True, bins = 5)
plt.show()





           #Outlier Checking Using Boxplot  (There seems to be some
outliers in this variable but they are all points of influence and
therefore no need to drop them)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.chol, color = 'mediumaquamarine',)
plt.show()


q1=211  ; q3=274.75
q1 - 1.5* (q3 - q1)
q3 + 1.5* (q3 - q1)
cv[(cv.chol > 370.375) | ( cv.chol < 115.375)]['chol']
#NB:Medically,Cholestoral Above 239 is High but as data Analyst, i give
results based on the data.
```

```
#             #Normality Test
# shapiro(cv.chol)




     ##FBS   (Categorical variable- Bimodal Distribution)

cv.describe()
cv.fbs .describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.fbs, kde = True, bins = 5,)
plt.show()




         #Outlier Checking Using Boxplot  (There seems to be an outlier in
this variable but they are all valid and therefore no need to drop them)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.fbs, color = 'mediumaquamarine',)
plt.show()







    ## RESTECG (Categorical variable- Multimodal Distribution)

cv.describe()
cv. restecg.describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.restecg, kde = True, bins = 5,axlabel = axes[2])
plt.show()




       #Outlier Checking Using Boxplot  (No outliers)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.restecg, color = 'mediumaquamarine',)
plt.show()
```

```
    ## THALACH  (Continuous variable- Unimodal Distribution ,follows
normal distribution)
cv.describe()
cv.thalach .describe()
cv.thalach .skew()


plt.figure(figsize= (10,5))
sns.distplot(x= cv.thalach, kde = True, bins = 5,)
plt.show()




     #Outlier Checking Using Boxplot (There seems to be some outliers in
this variable but they seems to be valid and therefore no need to treat
them)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.thalach, color = 'mediumaquamarine',)
plt.show()


q1=133.25  ; q3=166
q1 - 1.5* (q3 - q1)
q3 + 1.5* (q3 - q1)
cv[(cv.thalach > 215.125) | ( cv.thalach < 84.125)][['thalach','age']]


# cv['new_thalach'] = 220 - cv.age

# cv[['new_thalach','age']]

# cv[(cv.thalach > cv.new_thalach)][['thalach','new_thalach']]

# cv[(cv.thalach <= cv.new_thalach)][['thalach','new_thalach','age']]


#        #Normality Test
# shapiro(cv.thalach)





                   ## EXANG   (Categorical variable- Bimodal Distribution)

cv.describe()
cv.exang .describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.exang , kde = True, bins = 5,)
plt.show()




      #Outlier Checking Using Boxplot (No outliers)
plt.figure(figsize= (15, 5))
sns.boxplot(cv.exang , color = 'mediumaquamarine',)
plt.show()




    ##  OLDPEAK  (Continuous variable- Unimodal Distribution ,slightly
positively skewed but follows normal distribution)

cv.describe()
cv. oldpeak .describe()
cv.oldpeak.skew()

plt.figure(figsize= (10,5))
sns.distplot(x= cv. oldpeak, kde = True, bins = 5,)
plt.show()




     #Outlier Checking Using Boxplot (There seems to be some outliers in
this variable but they are all points of influence and therefore no need
to drop them)

plt.figure(figsize= (15, 5))
sns.boxplot(cv. oldpeak, color = 'mediumaquamarine',)
plt.show()


q1=0  ; q3=1.6
q1 - 1.5* (q3 - q1)
q3 + 1.5* (q3 - q1)
cv[(cv.oldpeak > 4) | ( cv.oldpeak < -2.4000000000000004)]['oldpeak']

#cv.oldpeak.loc[(cv.oldpeak > 4) | ( cv.oldpeak < -2.4000000000000004)] =
cv.oldpeak.mean()
#cv.drop(cv[(cv.oldpeak > 4) | ( cv.oldpeak < -
2.4000000000000004)].index,axis = 0,inplace = True)


#                #Normality Test
# from scipy.stats import normaltest
# shapiro(cv.oldpeak)
```

```
## SLOPE  (Categorical variable- Multimodal Distribution)

cv.describe()
cv.slope .describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.slope, kde = True, bins = 5,)
plt.show()




      #Outlier Checking Using Boxplot  (No outliers)
plt.figure(figsize= (15, 5))
sns.boxplot(cv.slope, color = 'mediumaquamarine',)
plt.show()






     ## CA  (Categorical variable- Multimodal Distribution)

cv.describe()
cv.ca .describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.ca, kde = True, bins = 5,)
plt.show()




     #Outlier Checking Using Boxplot  (There seems to be some outliers or
inappropriate data entries in this variable and needs to be treated as
such)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.ca, color = 'mediumaquamarine',)
plt.show()


#Treating outliers or inappropriate data entry
cv[(cv.ca > 3)['ca']

cv.ca.value_counts()

cv.ca.replace(to_replace = 4, value = 0,inplace = True)
```

```
    ## THAL  (Categorical variable- Multimodal Distribution)

cv.thal .describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.thal, kde = True, bins = 5,)
plt.show()



     #Outlier Checking and Inappropriate Data Entry Using Boxplot  (There
seems to be some Inappropriate Data Entry in this variable.The range is
supposed to be 1-3 according to the variable description but level 0 is
included and needs to be treated as such)

plt.figure(figsize= (15, 5))
sns.boxplot(cv.thal, color = 'mediumaquamarine',)
plt.show()


cv[(cv.thal < 1)]

cv.thal.value_counts()

cv.thal.replace(to_replace = 0, value = 2,inplace = True)



     ##TARGET   (Categorical variable- Multimodal Distribution)

cv.describe()
cv.target .describe()

plt.figure(figsize= (10,5))
sns.distplot(x= cv.target, kde = True, bins = 5,)
plt.show()



      #Outlier Checking Using Boxplot  (No outliers)
plt.figure(figsize= (15, 5))
sns.boxplot(cv.target, color = 'mediumaquamarine',)
plt.show()
```

```
#STATISTICAL SUMMARY USING PROFILE REPORT:

!pip install pandas-profiling

import pandas_profiling as pp

pp.ProfileReport(cv)

pr = pp.ProfileReport(cv, title = 'Cadio')
pr.to_file(output_file = 'Capstone.html')
```

```
#VARIABLES WHICH ARE CATEGORICAL IN ACTUAL SENSE BUT WRONGLY IMPORTED AS
NUMERIC


       #SEX(There are more males(206) than females(96) with respect to the
data)

sc = cv.sex.value_counts()
sns.countplot(y ='sex', data = cv, order=sc.index, palette= 'Set3')


cv['sex'] = cv['sex'].astype(str)
```

```
      #CP(Out of 302 patients; 143 have level 0 of chest pain, 86 have
level 2 chest pain,50 has level 1 chest pain and 23 have level 3 chest
pain)

cc = cv.cp.value_counts()
sns.countplot(y ='cp', data = cv, order=cc.index, palette= 'Set3')


cv['cp'] = cv['cp'].astype(str)
```

```
      #FBS(Out of the total numbrr of records here, 45 patients have their
fasting blood sugar > 120 mg/dl and the remaining 257 patients have their
fasting blood sugar <= 120 mg/dl)
```

```
fbs_c = cv.fbs.value_counts()
sns.countplot(y ='fbs', data = cv, order=fbs_c.index, palette= 'Set3')


cv['fbs'] = cv['fbs'].astype(str)




      #EXANG(99 of the patients have experienced exercise induced angina
and the other 203 have not)

exang_c = cv.exang.value_counts()
sns.countplot(y ='exang', data = cv, order=exang_c.index, palette= 'Set3')

cv['exang'] = cv['exang'].astype(str)




      #CA(175 patients have no major vessels coloured by flouroscopy,65
patients 1 major vessels coloured by flouroscopy, 38 patients have 2 major
vessels coloured and the remaining 24 patients also have 3 major vessels
coloured)

ca_c = cv.ca.value_counts()
sns.countplot(y ='ca', data = cv, order=ca_c.index, palette= 'Set3')


cv['ca'] = cv['ca'].astype(str)





      #THAL( 18 patients have normal thalaseemia; 165 patients also have
fixed defect and 119 patients have reversable defect)

thal_c = cv.thal.value_counts()
sns.countplot(y ='thal', data = cv, order=thal_c.index, palette= 'Set3')


cv['thal'] = cv['thal'].astype(str)




      #SLOPE(In levels,141 patients have slope 2, 140 have slope 1 and the
remaining 21 have slope 0)

slope_c = cv.slope.value_counts()
sns.countplot(y ='slope', data = cv, order=slope_c.index, palette= 'Set3')
```

```
cv['slope'] = cv['slope'].astype(str)






      #SLOPE(In levels,151 patients have restecg 2, 147 have restecg 0 and
the remaining 4 have restecg 2)

restecg_c = cv.restecg.value_counts()
sns.countplot(y ='restecg', data = cv, order=restecg_c.index, palette=
'Set3')


cv['restecg'] = cv['restecg'].astype(str)







      #TARGET(out of 302 patients, 164 of them have been targeted to have
cadiovascular disease and the remaining 138 don't have)

target_c = cv.target.value_counts()
sns.countplot(y ='target', data = cv, order=target_c.index, palette=
'Set3')


cv['target'] = cv['target'].astype(str)




cv.info()







#STUDYING THE OCCURENCE OF CVD ACROSS DIFFERENT AGE

            #A)Analysing Age Vrs CVD Using Displot and Histogram
sns.distplot(cv.loc[cv.target== '0', 'age'], color='forestgreen', hist =
False, label = 'target',)
sns.distplot(cv.loc[cv.target== '1', 'age'], color='orangered', hist =
False, label = 'target',)


sns.distplot(cv.loc[cv.target== '0', 'age'], color='forestgreen', hist =
True, label = 'target', )
```

```
sns.distplot(cv.loc[cv.target== '1', 'age'], color='orangered', hist =
True, label = 'target', )

cv[(cv.age > 58) & (cv.age <= 65)][['age','target']] --------------> #
Marjority of the non_targetted patients are within the class age of 58 to
65
cv[(cv.age > 50) & (cv.age <= 65)][['age','target']] --------------> #
Marjority of the targetted patients are within the class age of 50 to 55


#colors = ['forestgreen', 'orangered', 'steelblue', 'indigo',
'darkturquoise', 'yellowgreen']


            #B)Analysing Age Vrs CVD Using Boxplot

plt.figure(figsize = (10,5))
sns.boxplot(x = cv.age, y = cv.target,)
plt.show()



plt.figure(figsize = (10,5))
sns.boxplot(x = cv.age, y = cv.target, whis= 3,)
plt.show()



           #C)Analysing Age Vrs CVD Using Countplot


cv['bin_age'] = pd.cut(cv.age,bins = 4, right = False ,
include_lowest=True)

bin_c = cv.bin_age.value_counts()
sns.countplot(y ='bin_age', data = cv, order=bin_c.index, palette= 'Set3')



bin_c = cv.bin_age.value_counts()
sns.countplot(y ='bin_age', data = cv,hue = 'target', order=bin_c.index,
palette= 'Set3')



cv['bin_age2'] = pd.cut(cv.age,bins = 2, right = False ,
include_lowest=True)

bin_c2 = cv.bin_age2.value_counts()
sns.countplot(y ='bin_age2', data = cv,hue = 'target', order=bin_c2.index,
palette= 'Set3')
```

```
            #D)Analysing Age Vrs CVD Using ANOVA

                     #H0 : There is no significant effect between age and
CVD occurence


import statsmodels.api as sm
from statsmodels.formula.api import ols

#perform two-way ANOVA
model = ols('age ~ target', data=cv).fit()
sm.stats.anova_lm(model, typ=1)




             # E) ANALYSING AGE VRS TARGET USING CHI SQUARE

                     #H0 : There is no kind of relationship between age
and CVD occurence

observed_frequency = pd.crosstab(cv.bin_age2,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val)


                 # #OBSERVATIONS (AGE VRS CDV OCCURENCE)

                    # 1)According to the spread of the data, the targeted
CDV patients information gives more details(high variance or std and wider
range)
                    #    to our analysis across the various ages than that
of the non-targeted CVD patients which is even affected by an outlier.
```

```
                    # 2) At alpha = 0.05, There exist a significant effect
between different age and the occurence of CVD (P-value = 0.000104)

                    # 3)When Age is grouped into two namely 29 -53 and 53-
77, The occurence of CDV is more targeted towards the middle age(from age
29 to age 53) than that of Senior Years(age 53 - age 78)

                    # 4) At alpha = 0.05, There is a kind of relationship
between different age and the occurence of CVD (P- value =
0.0000237606464159380 )

                    # 5) Age group between 29 - 53 are more vulnerable to
heart attack..


#from scipy.stats import f_oneway
#f_oneway(cv.age,cv.target)







#DETECTING HEART ATTACK BASED ON ANOMALIES IN TRESTBPS

plt.figure(figsize= (10, 5))
sns.boxplot(cv.trestbps, color = 'mediumaquamarine',)
plt.show()


cv.trestbps.describe()

q1=120  ; q3=140
q1 - 1.5* (q3 - q1)
q3 + 1.5* (q3 - q1)
cv[(cv.trestbps > 170) | (cv.trestbps < 90)][['trestbps','target']]


                  #CONCLUSION ; At one point we can detect and at another
point we can not detect heart attack based on the anomalies
                  #  in resting blood pressure.. This is because the data
doesn't give us a convincing evidence to come to that logical conclusion.
It's a mixed answer,hence we can't detect.







#COMPOSITION OF OVERALL PATIENTS w.r.t GENDER(where Male = 1 and Female =
0)
```

```
gender_c = cv.sex.value_counts()

plt.figure(figsize = (12,5))
gender_c.plot.pie(radius = 1.2,
           autopct = '%1.2f %%',
           explode = [0.04,0.04], # specifies the distancce of the wedge
from the ccennter of the pie
           textprops = {'size' : 12, 'color' : 'steelblue'},
           wedgeprops = {'edgecolor' : 'white','width' :0.65 },
           cmap = 'Set3',
           shadow = True
                   )
plt.ylabel('')
plt.title('Pie Chart\n', size = 30, color = 'Purple', weight = 'bold')
plt.show()


gender_c.plot.pie()
gender_c.plot.barh()



                             #INPUT : There are more male patients (206)
than female patients(96).. The males constitute 68.21% of the records and
the renaining 31.79% are females
```

```
# RELATIONSHIP BETWEEN CHOLESTEROL LEVEL AND OCCURENCE OF CDV

      # ANALYSING CHOL VRS OCCURENCE OF CDV USING DISTPLOT & HISTOGRAM

sns.distplot(cv.loc[cv.target == '0', 'chol'], color='forestgreen', hist =
True, label = 'target', )
sns.distplot(cv.loc[cv.target == '1', 'chol'], color='orangered', hist =
True, label = 'target', )




      # ANALYSIS CHOL VRS CDV OCCURENCE USING BOXPLOT


plt.figure(figsize = (10,5))
sns.boxplot(x = cv.chol, y = cv.target,)
plt.show()


plt.figure(figsize = (10,5))
sns.boxplot(x = cv.chol, y = cv.target, whis = 3,)
```

```
plt.show()





      # ANALYSING CHOL AND CDV OCCURENCE USING COUNTPLOT

cv['bin_chol2'] = pd.cut(cv.chol,bins = 2, right = False ,
include_lowest=True)


chol_c2 = cv.bin_chol2.value_counts()
sns.countplot(y ='bin_chol2', data = cv,hue = 'target',
order=chol_c2.index, palette= 'Set3')
sns.countplot(y ='bin_chol', data = cv,hue = 'target', order=chol_c.index,
palette= 'Set3')





      # ANALYSING CHOL VRS TARGET USING ANOVA

             #H0 : There is no significant effect between CHOL AND TARGET

import statsmodels.api as sm
from statsmodels.formula.api import ols

                 #perform two-way ANOVA
model2 = ols('chol~target', data= cv).fit()
sm.stats.anova_lm(model2, typ=1)

#0.158037





       # ANALYSING CHOL AND CVD OCCURENCE USING PIE CHART

chol_c1 = cv.groupby('bin_chol2')['target'].value_counts()

plt.figure(figsize = (12,5))
chol_c1.plot.pie(radius = 1.2,
           autopct = '%1.2f %%',
           explode = [0.05,0.05,0.05,0.05], # specifies the distancce of
the wedge from the ccennter of the pie
           textprops = {'size' : 12, 'color' : 'steelblue'},
           wedgeprops = {'edgecolor' : 'white','width' :0.65 },
           cmap = 'Set2',
           shadow = True
                   )
```

```
plt.ylabel('')
plt.title('Pie Chart\n', size = 30, color = 'Blue', weight = 'bold')
plt.show()



    # ANALYSING CHOLESTEROL LEVEL VRS TARGET USING CHI SQUARE

                    #H0 : There is no kind of relationship between level
of cholestoral and CVD occurence

observed_frequency = pd.crosstab(cv.bin_chol2,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val)


                   # CONCLUSION :
                                  # 1) At alpha = 0.05, There is no
significant effect between cholestorol level and the occurence of CDV (P-
value = 0.158037)

                                  #  2) Patients in both cholestorol
groups(126 - 345 and 345 - 564) are slightly vulnerable to the
occurencence of CVD although Majority of the patients(97%) have their
cholestoral level to be between 126- 345.

                                  #  3) At alpha = 0.05, There is no kind
of relationship between the levels of cholesterol and the occurence of
heart attack




#RELATIONSHIP BETWEEN PEAK EXERCISING AND THE OCCURENCE OF HEART ATTACK
```

```
          # ANALYSING EXANG AND CDV OCCURENCE USING COUNTPLOT

exang_c = cv.exang.value_counts()
sns.countplot(y ='exang', data = cv,hue = 'target', order=exang_c.index,
palette= 'Set3')




          # ANALYSING EXANG VRS TARGET USING CHI SQUARE

                #H0 : There is no kind of relationship between PEAK
EXERCISE AND TARGET

observed_frequency = pd.crosstab(cv.exang,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.16f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.16f}'.format(p, alpha))



conclude(p_val) #Reject the Null with p-value = 0.0000000000000956




        # ANALYSING EXANG VRS CDV OCCURENCE USING PIE CHART

exang_c1 = cv.groupby('exang')['target'].value_counts()



plt.figure(figsize = (12,5))
exang_c1.plot.pie(radius = 1.2,
           autopct = '%1.2f %%',
           explode = [0.05,0.05,0.05,0.05], # specifies the distancce of
the wedge from the ccennter of the pie
           textprops = {'size' : 12, 'color' : 'steelblue'},
           wedgeprops = {'edgecolor' : 'white','width' :0.65 },
           cmap = 'Set2',
           shadow = True
                   )
plt.ylabel('')
plt.title('Pie Chart\n', size = 30, color = 'Blue', weight = 'bold')
plt.show()

# observed_frequency

# target   0    1
# exang
# 0       62  141
# 1       76   23



              # #CONCLUSION:

                  #   1)Those who do not engage in peak exercise are more
vulnerable to the disease.

                  #   2)Engaging in peak exercise does not 100% guarantee
a patients  from not having a heart attack although peak exercise is
helpful.

                  #   3)At alpha = 0.05, There exist a kind of
relationship between peak exercise and the occurence of CDV.

                  #   4)46.69% of the patients do not engage in peak
exercise and are vulnerable to heart attack.

                  #   5)25.17% of the patients engage in peak exercise and
are not vulnerable to CDV.

                  #   6)20.53% of the patients do not engage in peak
exercise yet, are not vulnerable to heart attack.

                  #   7)7.62% of the patients engage in peak exercise yet
are still vulnerable to occurence of CDV.


 #pd.crosstab(cv.exang,cv.target,normalize = 'index'/ normalize =
'column'/normalize = 'all')




# THAL VRS CVD


     # ANALYSING THAL AND CDV OCCURENCE USING COUNTPLOT
```

```
thal_c = cv.thal.value_counts()
sns.countplot(y ='thal', data = cv,hue = 'target', order=thal_c.index,
palette= 'Set3')




    # ANALYSING THAL VRS TARGET USING CHI SQUARE

               #H0 : Thalassemia is not a major cause of heart attack

observed_frequency = pd.crosstab(cv.thal,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #Reject the Null with p-value =  0.0000000000000000031




     # ANALYSING THAL AND OCCURENCE OF CDV USING PIE CHART

thal_c1 = cv.groupby('thal')['target'].value_counts()



plt.figure(figsize = (12,5))
thal_c1.plot.pie(radius = 1.2,
           autopct = '%1.2f %%',
           explode = [0.05,0.05,0.05,0.05,0.05,0.05,0.05,0.05], #
specifies the distancce of the wedge from the ccennter of the pie
           textprops = {'size' : 12, 'color' : 'steelblue'},
           wedgeprops = {'edgecolor' : 'white','width' :0.65 },
           cmap = 'Set2',
           shadow = True
                   )
```

```
plt.ylabel('')
plt.title('Pie Chart\n', size = 30, color = 'Blue', weight = 'bold')
plt.show()


# target   0    1
# thal
# 1       12    6
# 2       36  129
# 3       90   29


             # #OBSERVATIONS :

               #     1) At alpha = 0.05, Thalassemia is a major cause of
heart attack(P-value = 0.000000000000000000031)

               #     2)Patients with fixed defect thalassemia(level 2) are
more prone to heart attacks.

               #     3)Patients with normal defect(level 1) and reversable
defect(level 3) are less prone to the occurence of CDV.

               #     4)54.64% of the total number of patients have fixed
defect thalassemia.

               #     5)42.72% of the patients have fixed defect
thalassemia and are vulnerable to the disease
               #        whilst 11.92% have fixed defect thalassemia yet
are not vulnerable.

               #     6)39.40% of the total number of patients have
reversable defect thalassemia.

               #     7)9.60% of the patients have reversable defect
thalassemia and are vulnerable to the disease
               #        whilst 29.80% have reversable defect thalassemia
and are not vulnerable.

               #     8)5.96% of the total number of patients have normal
defect thalassemia.

               #     9)1.99% of the patients have normal defect
thalassemia and are vulnerable to the disease.
               #        whilst 3.97% have normal defect thalassemia and
are not vulnerable.




        #HOW ARE THE OTHER FACTORS DETERMINING THE OCCURENCE OF CVD?
```

```
                  #1) THALACH

                      #ANALYSING USING HEATMAP

cv['bin_thalach'] = pd.cut(cv.thalach,bins=2,right = False ,
include_lowest=True)
table = pd.crosstab(cv.target,cv.bin_thalach,normalize = 'columns')
table1 = pd.crosstab(cv.target,cv.bin_thalach)
table2 = pd.crosstab(cv.target,cv.bin_thalach,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.bin_thalach,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(table, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")




                        #ANALYSING USING ANOVA

                             #H0 : There is no significant effect between
THALACH AND TARGET

import statsmodels.api as sm
from statsmodels.formula.api import ols

#perform two-way ANOVA
model2 = ols('thalach~target', data= cv).fit()
sm.stats.anova_lm(model2, typ=1)


                          # ANALYSING THALACH VRS TARGET USING CHI SQUARE

               #H0 : There is no KIND OF RELATIONSHIP between THALACH AND
TARGET

observed_frequency = pd.crosstab(cv.bin_thalach,cv.target)
observed_frequency


import scipy.stats as stats
```

```
stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.16f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.16f}'.format(p, alpha))



conclude(p_val)

                                     #CONCLUSION :
                                         # 1) At alpha = 0.05, There is a
significant effect between thalach and the occurence of CVD (P-value =
2.476146e-14)
                                         # 2) Patients having thalach
within the range of 136 - 202 are more vulnerable to heart attack
                                         # 3) At alpha = 0.05, there is a
kind of relationship between different levels of thalach and the occurence
of CVD




              #2) OLDPEAK

                      #ANALYSING USING HEATMAP

cv['bin_oldpeak'] = pd.cut(cv.oldpeak,bins=2,right = False ,
include_lowest=True)
table1 = pd.crosstab(cv.target,cv.bin_oldpeak,normalize = 'columns')
table = pd.crosstab(cv.target,cv.bin_oldpeak)
table2 = pd.crosstab(cv.target,cv.bin_oldpeak,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.bin_oldpeak,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(table, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")
```

```
plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")



                      #ANALYSING USING ANOVA

                             #H0 : There is no significant effect between
THALACH AND TARGET

import statsmodels.api as sm
from statsmodels.formula.api import ols

#perform two-way ANOVA
model2 = ols('oldpeak~target', data= cv).fit()
sm.stats.anova_lm(model2, typ=1)



                    # ANALYSING OLDPEAK VRS TARGET USING CHI SQUARE

                     #H0 : There is no kind of relationship between
OLDPEAK and CVD occurence

observed_frequency = pd.crosstab(cv.bin_oldpeak,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val)


                                   #CONCLUSION :
                                         # 1) At alpha = 0.05, There is a
significant effect between oldpeak and the occurence of CVD (P-Value =
5.814567e-15 )
```

```
                                         # 2) Patients whose oldpeak are
within the range of 0 - 3.1 are more vulnerable to heart attack
                                         #3) At alpha = 0.05, There is a
kind of relationship between oldpeak and the occurence of CVD




              #3) SEX

                 # ANALYSING USING HEATMAP

observed_frequency = pd.crosstab(cv.target,cv.sex)
table1 = pd.crosstab(cv.target,cv.sex,normalize = 'columns')
table2 = pd.crosstab(cv.target,cv.sex,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.sex,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(observed_frequency, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")

plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGnBu")



                    # ANALYSING USING CHI SQUARE

                          # H0: There is no reltionship between a
patient's gender and occurence of CVD

observed_frequency = pd.crosstab(cv.sex,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)
```

```
def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #p = 0.0000015508552054950




                               #CONCLUSION :

                                         # 1)At alpha = 0.05, There is a
relationship between a patient's gender and the occurence of CVD (P-Value
= 0.0000015508552054950 )
                                         # 2) The possibility of a female
patient to be vulnerable to heart attack is very high. 75% of female
patients are vulnerable and the other 25% are not.





 #4) CP

                 # ANALYSING USING HEATMAP

observed_frequency = pd.crosstab(cv.target,cv.cp)
table1 = pd.crosstab(cv.target,cv.cp,normalize = 'columns')
table2 = pd.crosstab(cv.target,cv.cp,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.cp,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(observed_frequency, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="BrBG")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="BrBG")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="BrBG")

plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="BrBG")



                    # ANALYSING USING CHI SQUARE
```

```
                          # H0: There is no reltionship between a
patient's chest pain level and occurence of CVD

observed_frequency = pd.crosstab(cv.sex,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #p = 0.0000000000000000189




                               #CONCLUSION :

                                        # 1) At alpha = 0.05, There is a
relationship between a patient's chest pain level and the occurence of CVD
(P-Value = 0.0000000000000000189 )

                                        # 2) Patients with chest pain
level 1,level 2 and level 3 are vulnerable to heart attack. In fact, all
levels are vulnerable except level 0

                                        #3) Among the 3 levels which are
vulnerable, patients with level 2 chest pain are more vulnerable.





 #5) CA

                 # ANALYSING USING HEATMAP

observed_frequency = pd.crosstab(cv.target,cv.ca)
```

```
table1 = pd.crosstab(cv.target,cv.ca,normalize = 'columns')
table2 = pd.crosstab(cv.target,cv.ca,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.ca,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(observed_frequency, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="ocean")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True, cmap='ocean')

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True, cmap='ocean')

plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True, cmap='ocean')



                    # ANALYSING USING CHI SQUARE

                          # H0: There is no reltionship between a
patient's ca level and the occurence of CVD

observed_frequency = pd.crosstab(cv.ca,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #p = 7.50971406195956e-15



                                #CONCLUSION :
```

```
                                          # 1)At alpha = 0.05, There is a
relationship between a patient's ca level and the occurence of CVD (P-
Value = 0.0000000000000000189 )

                                          # 2) Patients with ca level 0 are
more vulnerable to heart attack. In fact, all levels are not vulnerable
except level 0
```

```
 #6) FBS

                 # ANALYSING USING HEATMAP

observed_frequency = pd.crosstab(cv.target,cv.fbs)
table1 = pd.crosstab(cv.target,cv.fbs,normalize = 'columns')
table2 = pd.crosstab(cv.target,cv.fbs,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.fbs,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(observed_frequency, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="BuPu")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="BuPu")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="BuPu")

plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="BuPu")



                    # ANALYSING USING CHI SQUARE

                          # H0: There is no relationship between a
patient's fasting blood sugar level and the occurence of CVD

observed_frequency = pd.crosstab(cv.fbs,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)
```

```
chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #p = 0.7611374700928197




                                #CONCLUSION :

                                          # 1)At alpha = 0.05, There is no
kind of relationship between a patient's fasting blood sugar level and the
occurence of CVD (P-Value =  0.7611374700928197, )
                                          # 2)Patients with fbs greater
than 120 are vulnerable somehow same as those with fbs less than or equal
to 120 but this variable does not convincingly come to a solid
conclusion.. It's a 50 : 50 affair..







 #7) RESTECG

                 # ANALYSING USING HEATMAP

observed_frequency = pd.crosstab(cv.target,cv.restecg)
table1 = pd.crosstab(cv.target,cv.restecg,normalize = 'columns')
table2 = pd.crosstab(cv.target,cv.restecg,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.restecg,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(observed_frequency, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="YlGn")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="YlGn")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="YlGn")
```

```
plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="YlGn")



                    # ANALYSING USING CHI SQUARE

                          # H0: There is no reltionship between a
patient's restecg level and the occurence of CVD

observed_frequency = pd.crosstab(cv.restecg,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)



def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #p = 0.0077130532693189778




                               #CONCLUSION :

                                         # 1) At alpha = 0.05, There is a
relationship between a patient's restecg level and the occurence of CVD
(P-Value = 0.0077130532693189778 )
                                         # 2) Patients with restecg level
1 are more vulnerable to heart attack. In fact, all levels are less
vulnerable except level 1






 #8) SLOPE

                 # ANALYSING USING HEATMAP
```

```
observed_frequency = pd.crosstab(cv.target,cv.slope)
table1 = pd.crosstab(cv.target,cv.slope,normalize = 'columns')
table2 = pd.crosstab(cv.target,cv.slope,normalize = 'index')
table4 = pd.crosstab(cv.target,cv.slope,normalize = 'all')


plt.figure(figsize=(12,8))
sns.heatmap(observed_frequency, vmin = -1, vmax = 1,fmt = "",annot=True,
cmap="OrRd")

plt.figure(figsize=(12,8))
sns.heatmap(table1, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="OrRd")

plt.figure(figsize=(12,8))
sns.heatmap(table2, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="OrRd")

plt.figure(figsize=(12,8))
sns.heatmap(table4, vmin = -1, vmax = 1,fmt = "",annot=True, cmap="OrRd")



                    # ANALYSING USING CHI SQUARE

                          # H0: There is no reltionship between a
patient's slope level and the occurence of CVD

observed_frequency = pd.crosstab(cv.slope,cv.target)
observed_frequency


import scipy.stats as stats

stats.chi2_contingency(observed_frequency)


chi2, p_val , df, expected_frequency =
stats.chi2_contingency(observed_frequency)




def conclude(p, alpha = 0.05):
    if p < alpha :
        print('Reject the Null with p = {:.19f}'.format(p))
    else :
        print('Fail to Reject the Null with p = {:.19f}'.format(p, alpha))



conclude(p_val) #p = 6.5777827609179e-11
```

```
                             #CONCLUSION :

                                        # 1) At alpha = 0.05, There is a
relationship between a patient's slope level and the occurence of CVD (P-
Value = 6.5777827609179e-11 )
                                        # 2) Patients with slope level 2
are more vulnerable to heart attack.
```

```
# UNDERSTANDING THE RELATIONSHIP BETWEEN ALL THE GIVEN VARIABLES USING
PAIR PLOT

sns.pairplot(data = cv,vars =
{'age','trestbps','chol','chol','exang','thal','sex','cp','fbs','restecg',
'thalach','oldpeak','slope','ca'})
sns.pairplot(data = cv, hue ='target')
sns.pairplot(data = cv)
```

```
#EXPORTING THE PROCESSED DATA:
cv.to_csv(r"\Users\oseia\OneDrive\Desktop\Python\cvd.csv")
```

```
#CREATING A COPY OF THE DATAFRAME:

new = cv.copy()


# new['target'] = new['target'].astype(int)
# cv['sex'] = cv['sex'].astype(str)
# cv['cp'] = cv['cp'].astype(str)
# cv['fbs'] = cv['fbs'].astype(str)
# cv['exang'] = cv['exang'].astype(str)
# cv['ca'] = cv['ca'].astype(str)
# cv['thal'] = cv['thal'].astype(str)
# cv['slope'] = cv['slope'].astype(str)
# cv['restecg'] = cv['restecg'].astype(str)
# cv['target'] = cv['target'].astype(str)
```

```
#PERFORMING LOGISTIC REGRESSION:
new1 = pd.get_dummies(new)
```

```
new1.info()


      #SPLITTING THE DATASET

from sklearn.model_selection import train_test_split as split

train, test = split(new1, test_size = 0.30, random_state = 12)


      #IMPORTING AND BUILDING THE MODEL USING LOGISTIC REGRESSION
from sklearn.linear_model import LogisticRegression



lr = LogisticRegression(max_iter = 5000)
lr.fit(train.drop(columns='target'), train.target)


# ## PREDICTION OF CLASSES AND PROBABILITY

# In[94]:


prob = lr.predict_proba(test.drop(columns='target'))
prob


# In[95]:


pred = lr.predict(test.drop(columns='target'))
pred




# ACCURACY, PRECISION , RECALL AND F1 SCORE



from sklearn import metrics

pd.crosstab(test.target, pred) # Confusion Matrix


metrics.accuracy_score(y_true=test.target, y_pred = pred)* 100 =
86.81318681318682


metrics.precision_score(y_true=test.target, y_pred = pred)* 100 =  92.5
metrics.recall_score(y_true=test.target, y_pred = pred)* 100 =
80.43478260869566

metrics.f1_score(y_true=test.target, y_pred = pred)* 100 =
86.04651162790698
```

                         #PREDICTION CONCLUSION :

                             # The accuracy,precision,recall and f1 scores suggest to us that the data is rich enough to predict the model.

```
# cv1 = cv.copy()

# cv1['sex'] = cv1['sex'].astype(int)
# cv1['cp'] = cv1['cp'].astype(int)
# cv1['fbs'] = cv1['fbs'].astype(int)
# cv1['exang'] = cv1['exang'].astype(v)
# cv1['ca'] = cv1['ca'].astype(int)
# cv1['thal'] = cv1['thal'].astype(int)
# cv1['slope'] = cv1['slope'].astype(int)
# cv1['target'] = cv1['target'].astype(int)
# cv1['restecg'] = cv1['restecg'].astype(int)


#cv1.drop(columns = [['fbs','chol']])
```