**Predicting Breast cancer:**

import numpy as np

import pandas as pd

%matplotlib inline

import matplotlib.pyplot as plt

import seaborn as sns

data = pd.read\_csv('C:/Users/Lenovo/Desktop/All Folders/breast cancer/CancerDataset.csv')

data.head()

data.shape

data.isna().sum()

data = data.dropna(axis = 1)

data.shape

data['diagnosis'].value\_counts()

sns.countplot(data['diagnosis'],label = "Count")

data.dtypes

from sklearn.preprocessing import LabelEncoder

LE = LabelEncoder()

data.iloc[:,1] = LE.fit\_transform(data.iloc[:,1].values)

data.head()

sns.pairplot(data, hue="diagnosis")

data.corr()

plt.figure(figsize=(20,20))

sns.heatmap(data.corr(), annot=True, fmt='.0%',cmap="BuPu")

X = data.iloc[:,2:3].values

Y = data.iloc[:, 1].values

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y,test\_size = 0.25,random\_state=0)

from sklearn.preprocessing import StandardScaler

ss = StandardScaler()

X\_train = ss.fit\_transform(X\_train)

X\_test = ss.transform(X\_test)

def MLmodels(X\_train,Y\_train):

from sklearn.linear\_model import LogisticRegression

LG = LogisticRegression(random\_state=0)

LG.fit(X\_train,Y\_train)

LG.predict(X\_test)

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n\_neighbors=5,metric='minkowski',p=2)

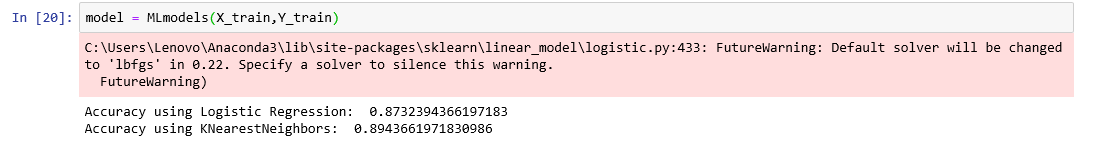
knn.fit(X\_train,Y\_train)

print('Accuracy using Logistic Regression: ',LG.score(X\_train,Y\_train))

print('Accuracy using KNearestNeighbors: ',knn.score(X\_train,Y\_train))

return LG,knn

model = MLmodels(X\_train,Y\_train)



from sklearn.metrics import confusion\_matrix

for i in range(len(model)):

c = confusion\_matrix(Y\_test,model[i].predict(X\_test))

TN = c[0][0]

TP = c[1][1]

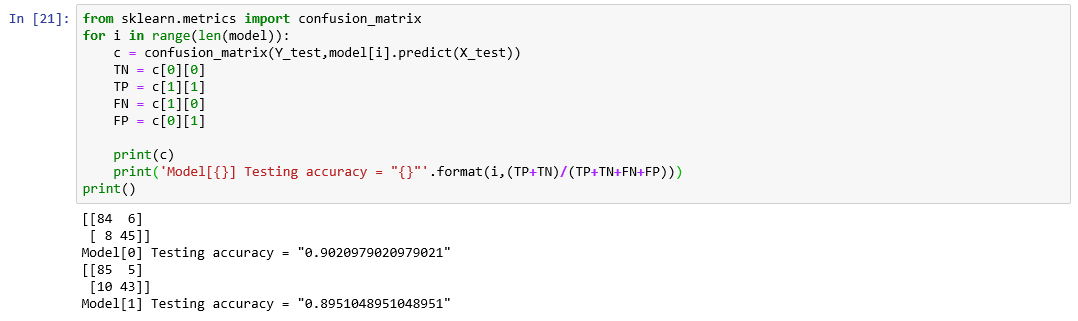
FN = c[1][0]

FP = c[0][1]

print(c)

print('Model[{}] Testing accuracy = "{}"'.format(i,(TP+TN)/(TP+TN+FN+FP)))

print()



from sklearn.metrics import classification\_report

from sklearn.metrics import accuracy\_score

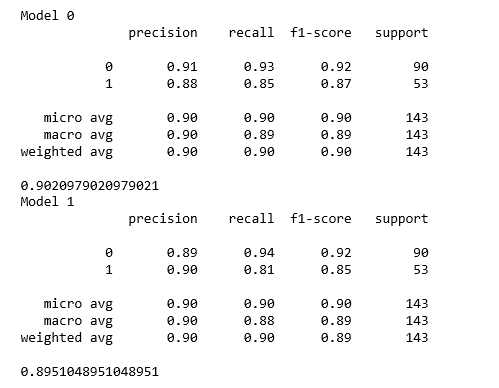
for i in range(len(model)):

print('Model',i)

print(classification\_report(Y\_test,model[i].predict(X\_test)))

print(accuracy\_score(Y\_test,model[i].predict(X\_test)))

print()



**Breast cancer Risk Analysis:**

import numpy as np

import pandas as pd

%matplotlib inline

import matplotlib.pyplot as plt

import seaborn as sns

import sklearn

data = pd.read\_csv('C:/Users/Lenovo/Desktop/All Folders/breast cancer/CancerDataset.csv')

data.head()

data.shape

data['city'].value\_counts()

sns.countplot(data['diagnosis'],label = "Count")

X = data.iloc[:,2:3].values

Y = data.iloc[:,1].values

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y,test\_size = 0.25,random\_state=0)

from sklearn.preprocessing import StandardScaler

ss = StandardScaler()

X\_train = ss.fit\_transform(X\_train)

X\_test = ss.transform(X\_test)

def MLmodels(X\_train,Y\_train):

from sklearn.linear\_model import LogisticRegression

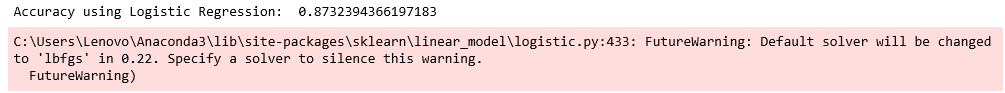
LG = LogisticRegression(random\_state=0)

LG.fit(X\_train,Y\_train)

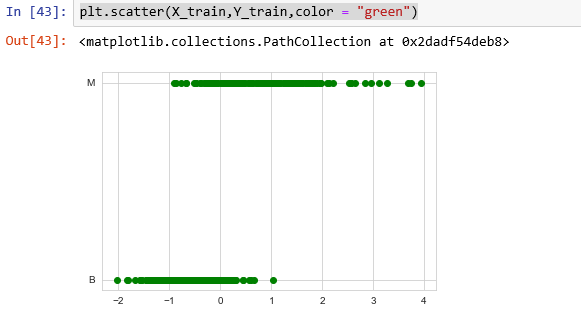
print('Accuracy using Logistic Regression: ',LG.score(X\_train,Y\_train))

return LG

model = MLmodels(X\_train,Y\_train)



plt.scatter(X\_train,Y\_train,color = "green")



df = pd.DataFrame(data)

from sklearn.preprocessing import LabelEncoder

labelencoder\_Y = LabelEncoder()

Y = labelencoder\_Y.fit\_transform(data['diagnosis'])

data['diagnosis'].value\_counts()



print(df[['diagnosis','city']])

dat = df[df['diagnosis'] == 'M']

dat.groupby(df['city']).count()

frame = pd.DataFrame(df['diagnosis']=='M').groupby(df['city']).count()

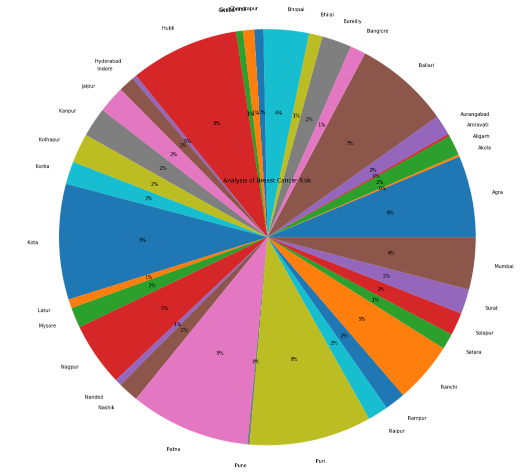
labels = 'Agra','Akola','Aligarh','Amravati','Aurangabad','Ballari','Banglore','Bareilly','Bhilai','Bhopal','Chandrapur','Gadhchiroli','Gwalior','Hubli','Hyderabad','Indore','Jaipur','Kanpur','Kolhapur','Korba','Kota','Latur','Mysore','Nagpur','Nanded','Nashik','Patna','Pune','Puri','Raipur','Rampur','Ranchi','Satara','Solapur','Surat','Mumbai'

#plt.figure(figsize=(15,14))

plt.title("Analysis of Breast Cancer Risk")

plt.pie(frame,labels=labels,autopct='%1.0f%%',radius=5)

#plt.legend()



**Analysis of lung cancer:**

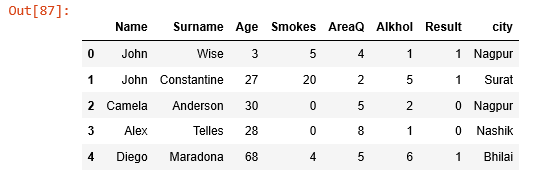
import numpy as np

import matplotlib.pyplot as plt

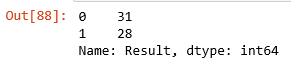
import pandas as pd

data = pd.read\_csv('C:/Users/Lenovo/Desktop/lung.csv')

data.head()

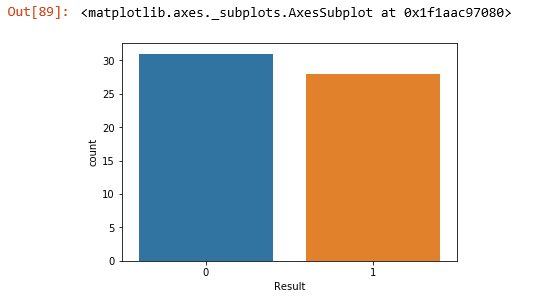


data['Result'].value\_counts()



import seaborn as sns

sns.countplot(data['Result'],label = "Count")



X = data.iloc[:,2:6].values

Y = data.iloc[:, 6].values

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y,test\_size = 0.25,random\_state=0)

from sklearn.preprocessing import StandardScaler

ss = StandardScaler()

X\_train = ss.fit\_transform(X\_train)

X\_test = ss.transform(X\_test)

def MLmodels(X\_train,Y\_train):

from sklearn.linear\_model import LogisticRegression

LG = LogisticRegression(random\_state=0)

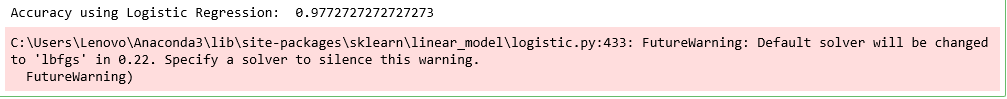
LG.fit(X\_train,Y\_train)

LG.predict(X\_test)

print('Accuracy using Logistic Regression: ',LG.score(X\_train,Y\_train))

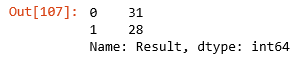
return LG

model = MLmodels(X\_train,Y\_train)

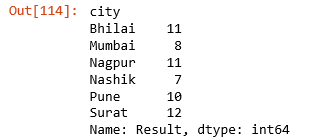


frame = pd.DataFrame(data[['Smokes','Result','city']])

data['Result'].value\_counts()



(frame['Result'] == 1).groupby(frame['city']).count()



f = pd.DataFrame((frame['Result'] == 1).groupby(frame['city']).count())

lbs = 'Bhilai','Mumbai','Nagpur','Nashik','Pune','Surat'

c = '#ff9999','#66b3ff','#99ff99','#ffcc99','#cc66ff','#ff66cc'

plt.title("Analysis of lung Cancer Risk Areawise")

plt.pie(f,labels=lbs,colors=c,autopct='%1.0f%%',radius=3)

centre\_circle = plt.Circle((0,0),0.90,fc='white')

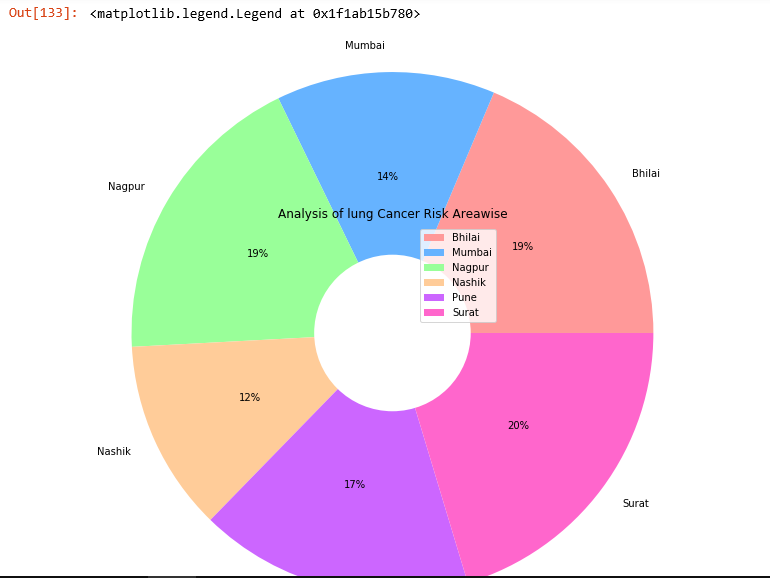
fig = plt.gcf()

fig.gca().add\_artist(centre\_circle)

#plt.axis('equal')

plt.tight\_layout()

plt.legend()



data = [31,28]

labels = ['Affected','Non-affected']

colors = ['#ff9999','#ffcc99']

plt.title('Percentage of People affected by lungcancer')

plt.pie(data,labels=labels,colors = colors,autopct='%1.0f%%',radius=3)

plt.legend()

