**1 and 4**

import numpy as np

import csv

import pandas as pd

from pgmpy.models import BayesianNetwork

from pgmpy.estimators import MaximumLikelihoodEstimator

from pgmpy.inference import VariableElimination

heartDisease = pd.read\_csv('heart.csv')

heartDisease = heartDisease.replace('?',np.nan)

model=BayesianNetwork([('age','trestbps'),('age','fbs'),

('sex','trestbps'),('exang','trestbps'),('trestbps','heartdisease'),('fbs','heartdisease'),('heartdisease','restecg'),

('heartdisease','thalach'),('heartdisease','chol')])

model.fit(heartDisease,estimator=MaximumLikelihoodEstimator)

print(model.get\_cpds('age')) **# Till here 1st one and the whole for 4th one**

HeartDisease\_infer = VariableElimination(model)

print('\n 1. Probability of HeartDisease given Age=34')

q=HeartDisease\_infer.query(variables=['heartdisease'],evidence={'age':34})

print(q)

print('\n 2. Probability of HeartDisease given cholesterol=210')

q=HeartDisease\_infer.query(variables=['heartdisease'],evidence={'chol':210})

print(q)

**Output**

+---------+------------+

| age(29) | 0.00390244 |

+---------+------------+

| age(34) | 0.00585366 |

+---------+------------+

| age(35) | 0.0146341  |

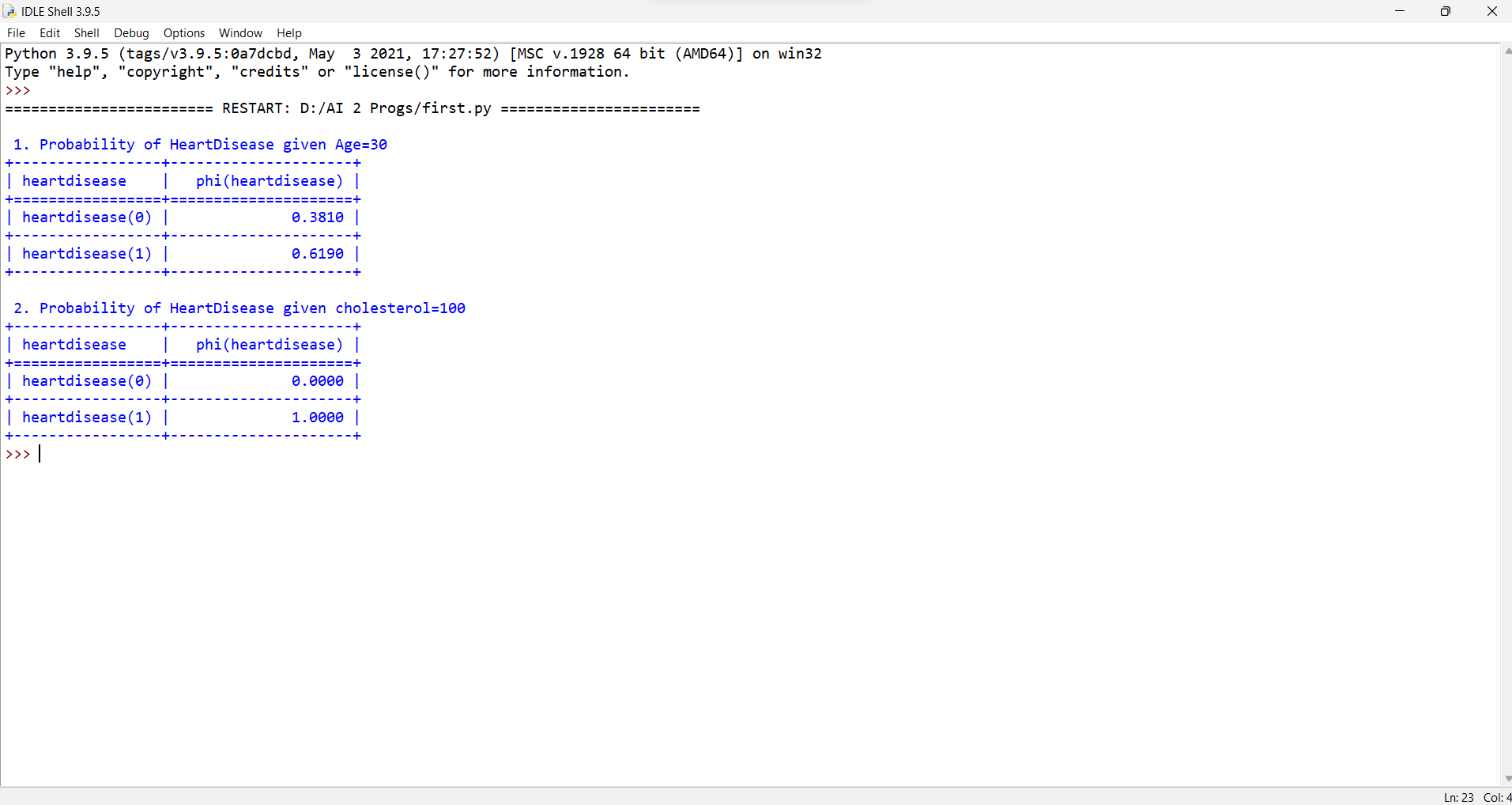
+---------+------------+

| age(37) | 0.00585366 |

+---------+------------+

| age(38) | 0.0117073  |

+---------+------------+



**5.**

import pandas as pd

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

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

from sklearn import tree

from dtreeviz.trees import\*

data = pd.read\_csv('diabetes.csv')

data.index+=1

data.head()

X = data.drop(columns='Outcome')

Y = data['Outcome']

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y,test\_size=0.3,random\_state=123)

clf= DecisionTreeClassifier(criterion='entropy',max\_depth=5, random\_state=0)

clf.fit(X\_train, Y\_train)

Y\_pred= clf.predict(X\_test)

print("Accuracy: ",accuracy\_score(Y\_test, Y\_pred))

feature\_cols = ['Pregnancies','Glucose' ,'BloodPressure', 'SkinThickness','Insulin', 'BMI', 'DiabetesPedigreeFunction','Age']

text\_representation = tree.export\_text(clf,feature\_names=feature\_cols)

print(text\_representation)

**Output**

**Output: Accuracy: 0.7705627705627706**

**A screenshot of a computer

Description automatically generated**

**3.**

import pandas as pd

from sklearn import preprocessing

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

from sklearn.naive\_bayes import GaussianNB

from sklearn import metrics

d=pd.read\_csv('Dataset.csv')

data=d.drop(columns='AGE')

le = preprocessing.LabelEncoder()

gdr,smk,alrgy,whzng,alcl,cough,sht\_of\_brth,chst\_pain,lung\_cncr=le.fit\_transform(data.GENDER),le.fit\_transform(data.SMOKING),le.fit\_transform(data.ALLERGY),le.fit\_transform(data.WHEEZING),le.fit\_transform(data.ALCOHOL\_CONSUMING),le.fit\_transform(data.COUGHING),le.fit\_transform(data.SHORTNESS\_OF\_BREATH),le.fit\_transform(data.CHEST\_PAIN),le.fit\_transform(data.LUNG\_CANCER)

f=list(zip(gdr,smk,alrgy,whzng,alcl,cough,sht\_of\_brth,chst\_pain))

x\_train, x\_test, y\_train, y\_test = train\_test\_split(f, lung\_cncr, test\_size=0.3,random\_state=0)

gnb = GaussianNB()

gnb.fit(x\_train, y\_train)

y\_pred=gnb.predict(x\_test)

print(“Lung cancer prediction”)

print("Accuracy for Naive Bayes : ",metrics.accuracy\_score(y\_test, y\_pred))

**Output**

**Accuracy for Naive Bayes : 0.8064516129032258**

**6.**

from pgmpy.models import BayesianNetwork

from pgmpy.factors.discrete import TabularCPD

import networkx as nx

import pylab as plt

from pgmpy.inference import VariableElimination

model = BayesianNetwork([('Guest', 'Host'), ('Price', 'Host')])

cpd\_guest = TabularCPD('Guest', 3, [[0.33], [0.33], [0.33]])

cpd\_price = TabularCPD('Price', 3, [[0.33], [0.33], [0.33]])

cpd\_host = TabularCPD('Host', 3, [[0, 0, 0, 0, 0.5, 1, 0, 1, 0.5],[0.5, 0, 1, 0, 0, 0, 1, 0, 0.5],[0.5, 1, 0, 1, 0.5, 0, 0, 0, 0]],evidence=['Guest', 'Price'], evidence\_card=[3, 3])

model.add\_cpds(cpd\_guest, cpd\_price, cpd\_host)

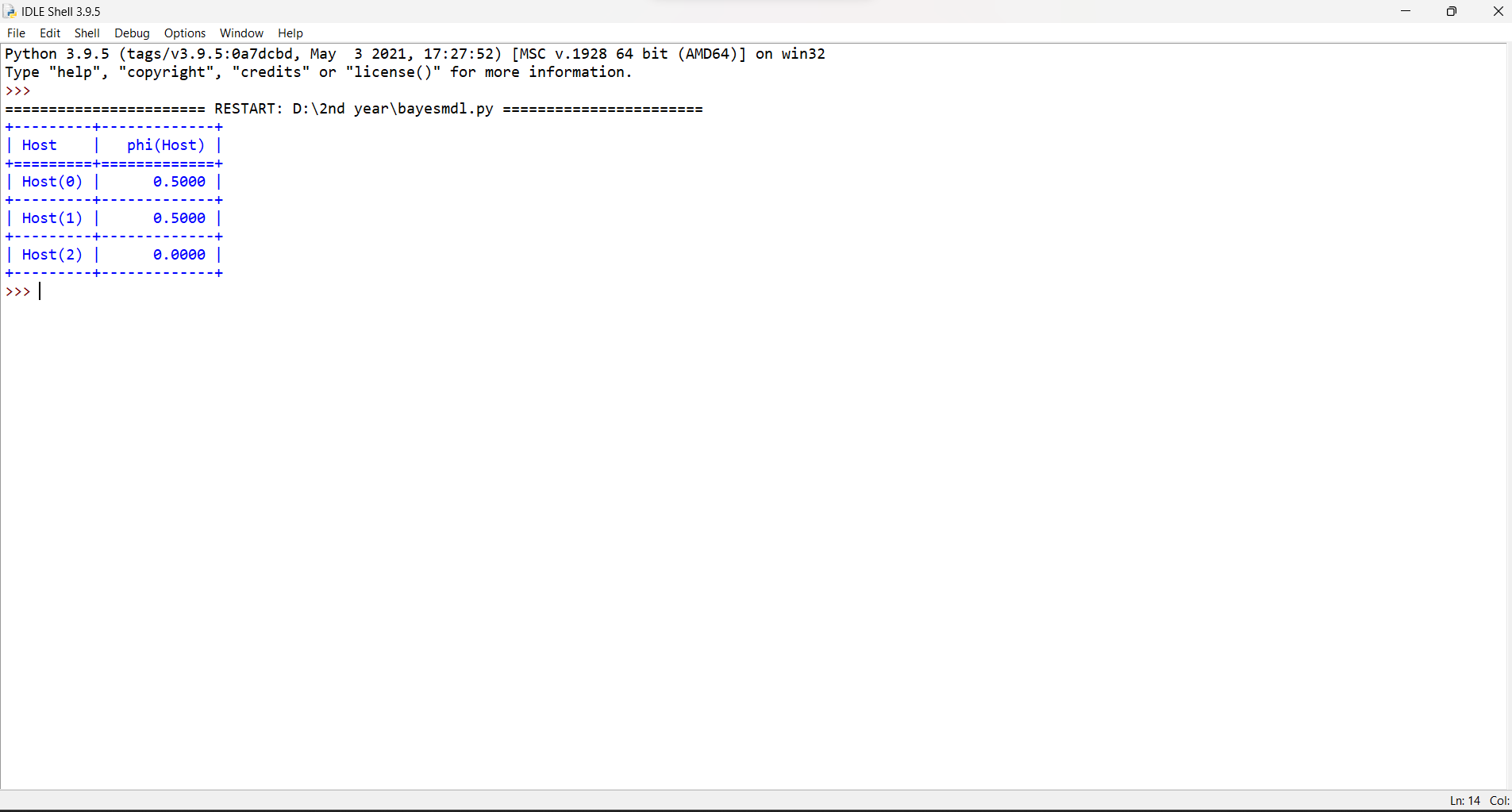
model.check\_model()

infer = VariableElimination(model)

posterior\_p = infer.query(['Host'], evidence={'Guest': 2, 'Price': 2})

print(posterior\_p)

**Output**

****

**2.**

from math import sin, pi

import numpy as np

from numpy.random import normal

from matplotlib import pyplot

def objective(x, noise=0.1):

noise = normal(loc=0, scale=noise)

return (x \*\* 2 \* sin(5 \* pi \* x) \*\* 6.0) + noise

X = np.arange(0, 1, 0.01)

y = [objective(x, 0) for x in X]

ynoise = [objective(x) for x in X]

ix = np.argmax(ynoise)

print(f'Optima: x={X[ix]:.3f}, y={ynoise[ix]:.3f}')

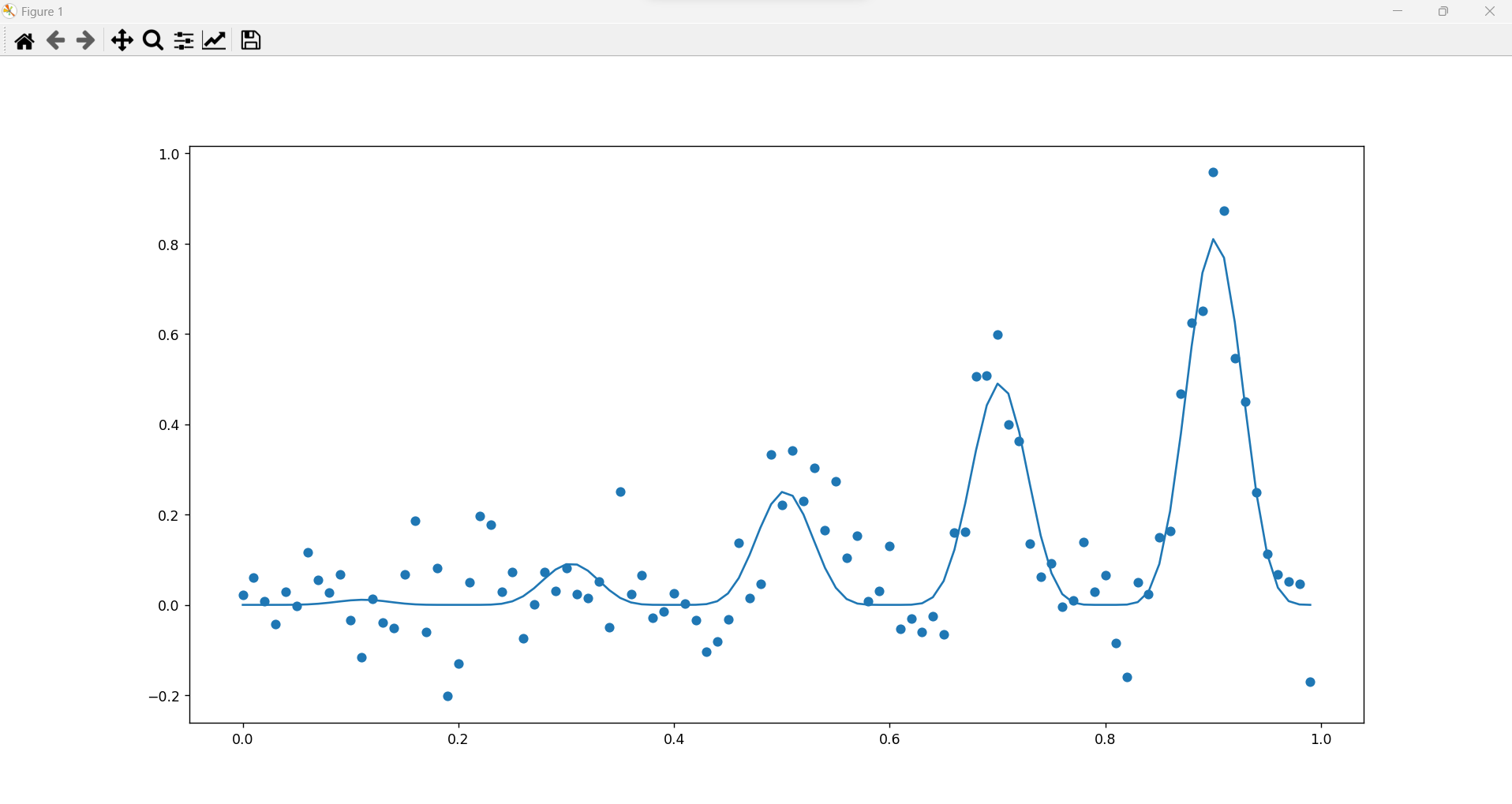
pyplot.scatter(X, ynoise)

pyplot.plot(X, y)

pyplot.show()

**Output**

**Optima: x=0.900, y=0.958**

****