Calculation

Weight(x)	height(y)	Chest size(z)	TShirt Size
55	165	45	S
67	152	49	M
70	169	47	M
80	174	49	L
88	175	50	XL
67	171	44	M
50	166	40	S
90	178	52	XL
98	175	50	XL

$d = \sqrt{(x-a)^2 + (y-b)^2 + (z-c)^2}$

$d = \sqrt{(55-80)^2 + (165-176)^2 + (45-50)^2} = 25.98$

$d = \sqrt{(165-80)^2 + (172-176)^2 + (49-50)^2} = 15.1$

$d = \sqrt{(80-80)^2 + (174-176)^2 + (49-50)^2} = 4.12$

$d = \sqrt{(88-80)^2 + (175-176)^2 + (50-50)^2} = 9.14$

$d = \sqrt{(67-80)^2 + (171-176)^2 + (44-50)^2} = 14.3$

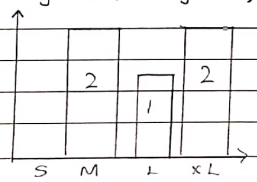
$d = \sqrt{(50-80)^2 + (166-176)^2 + (40-50)^2} = 81$

$d = \sqrt{(90-80)^2 + (178-176)^2 + (52-50)^2} = 12.9$

$d = \sqrt{(98-80)^2 + (175-176)^2 + (50-50)^2} = 18.6$

$d = \sqrt{(70-80)^2 + (169-176)^2 + (47-50)^2} = 10.4$

Input: a = weight = 80, b = height = 176, chestsize = 50. Let's set k value as



Write a program to implement K-nearest neighbours algorithm to classify the iris data set. Print both correct and wrong predictions. Java/python ML library classes can be used for this problem.

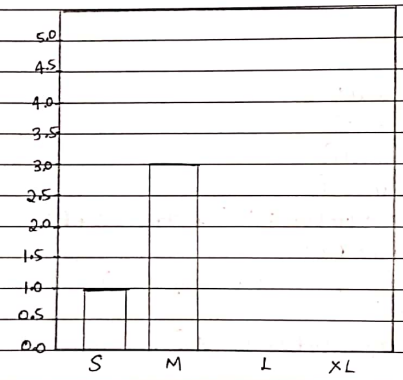
```
import csv
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
def KNN(datax):
    print("Enter your weight, height and chest size to buy right size")
    a,b,c = input().split()
    a = int(a)
    b = int(b)
    c = int(c)
    for line in datax:
        x = int(line[0])
        y = int(line[1])
        z = int(line[2])
        dist = np.sqrt((x-a)**2 + (y-b)**2 + (z-c)**2)
        line.append(dist)
    print("How many nearest survey - K = ")
    k = int(input())
    datax.sort(key = lambda i: i[4])
    scount = 0
    mcount = 0
    lcount = 0
    xlcount = 0
    for i in range(k):
        print(datax[i])
```

Output:

Enter your Weight, height and chest size to buy the right size
60 176 38

How many nearest survey K=
4

['55', '170', '45', 'S', 10.488 --]
['67', '171', '44', 'M', 10.488 --]
['66', '168', '46', 'M', 12.80 --]
['67', '172', '49', 'M', 13.63 ---]



--- GO FOR MEDIUM SIZE ---


```
if . datax[j][3] =='S':
    Scount +=1
if datax[j][3] =='M':
    Mcount +=1
if datax[j][3] =='L':
    lcount +=1
if datax[j][3] =='xl':
    xlcount +=1
if (Scount > mcount and Scount > lcount and Scount > xlcount):
    print ("---- GO SMALL SIZE ----")
elif (mcount > Scount and mcount > lcount and mcount > xlcount):
    print ("----GO MEDIUM SIZE-----")
elif (lcount > Scount and lcount > mcount and lcount > xlcount):
    print ("----GO LARGE SIZE------")
else:
    print ("----- Go XL SIZE -----")
def main():
    file = r"C:\Users\AMCEC\Desktop\T-Shirt.csv"
    data = pd.read_csv(file)
    display(data)
    fd = csv.reader(open(file))
    datax = []
    for line in fd:
        datax.append(line)
    knn(datax[1:])
main()
```

Problem Calculation.

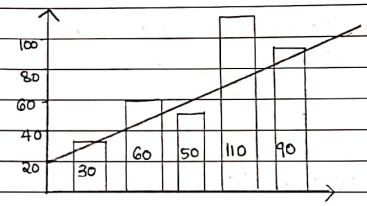
A cricketer

X	Y
1	30
2	60
3	50
4	110
5	90

$$M = \frac{\sum_{i=1}^{n}(xi-mx)(yi-my)}{\sum_{i=1}^{n}(xi-mx)^2}$$

$my = M * mx + C$

$C = my - M * mx$



$n = 5$

$\Sigma x = 15$

$\Sigma Y = 340$

$mx = \frac{\Sigma x}{n}$ $my = \frac{\Sigma y}{n}$

$= 15/5 = 3$ $= 340/5 = 68$

x	Y	(xi-mx)	(yi-my)	(xi-mx) * (yi-my)	(xi-mx)^2
1	30	(1-3)=-2	(30-68)=-38	(1-3) * (30-68)=76	4
2	60	-1	-8	8	1
3	50	0	-18	0	0
4	110	1	42	42	1
5	90	2	22	44	4
				170	10

Implement the non-parametric locally weighted regression algorithm in order to fit data points. Select appropriate data set for your experiment and draw graphs.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import csv
def reglne (x,y):
  plt.bar(x,y,label="AMCEC admission")
  plt.xlabel('Year')
  plt.ylabel('nos')
  plt.scatter(x,y,c='red')
  plt.show()
  mx=np.mean(x)
  my=np.mean(y)
  n=len(x)
  up=0
  M=0
  dw=0
  for in range(n):
    up+=(x[i]-mx)*(y[i]-my)
    dw+=(x[i]-mx)**2
  M=up/dw
  C=my-(M*mx)
  print("linear regression slope=",M)
  print("linear regression constant"=,c)
  max_x=np.max(x)+1
  min_x=np.min(x)-1
  x1=0
```

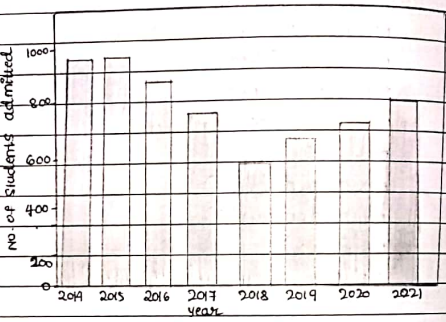
$M = 170/10 = 17$

$C = my - M * mx$
$= 68 - 17 * 3$
$= 68 - 51$
$= 17$

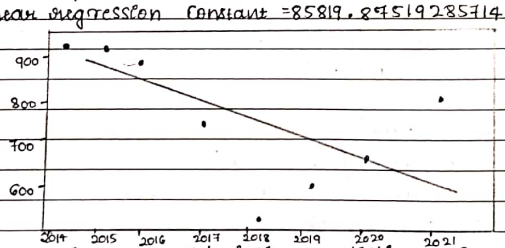
$y = M * x_i + C$
$y = 17 * 6 + 17$
$y = 119$

Output:

	Year	nos
0	2014	955
1	2015	957
2	2016	914
3	2017	737
4	2018	533
5	2019	595
6	2020	670
7	2021	820



linear regression slope: -42.15476190476190 5
linear regression constant = 85819.87519285714



Enter which year admission prediction : 2019
Predicted admission = 709.3928571428551


```
x1 = np.linspace(min_x, max_x, 6)
y1 = M*x1 + C
plt.plot(x1, y1, color='blue')
print(x)
plt.scatter(x,y,c='red')
plt.show()
print("Enter which year admission prediction")
year = int(input())
adm = M*year+C
print("predicted admission =",adm)
def main():
    file = r"C:\Users\AMC\Desktop\DATASET\AMCEC.csv"
    data = pd.read_csv(file)
    display(data)
    x = data['year'].values
    y = data['nos'].values
    regline(x,y)
main()
```

Calculation

n = 9

n(yes) = 5 n(no) = 4 P(yes) = 5/9 P(no) = 4/9

P(yes/(red, sports, domestic))

$= P(\text{red}/\text{yes}) * P(\text{sports}/\text{yes}) * P(\text{domestic}/\text{yes}) * P(\text{yes})$

P(no/(red, sports, domestic))

$= P(\text{red}/\text{no}) * P(\text{sports}/\text{no}) * P(\text{domestic}/\text{no}) * P(\text{no})$

Attribute = red

$n(\text{red}/\text{yes}) = 3$ $n(\text{red}/\text{no}) = 2$ $P(\text{red}/\text{yes}) = 3/5$ $P(\text{red}/\text{no}) = 2/4$

Attribute = sports

$n(\text{sports}/\text{yes}) = 3$ $n(\text{sports}/\text{no}) = 1$ $P(\text{sports}/\text{yes}) = 3/5$ $P(\text{sports}/\text{no}) = 1/4$

Attribute = domestic

$n(\text{domestic}/\text{yes}) = 3$ $n(\text{domestic}/\text{no}) = 2$ $P(\text{domestic}/\text{yes}) = 3/5$ $P(\text{domestic}/\text{no}) = 2/4$

P(yes) = 5/9 P(no) = 4/9

$P(\text{yes}/(\text{red, sports, domestic})) = 9/5 * 3/5 * 3/5 * 5/9$

$= 3/25$

$x = 0.12$

$P(\text{no}/(\text{red, sports, domestic})) = 2/4 * 1/4 * 2/4 * 4/9$

$= 1/36$

$y = 0.027$

% car stolen $= \frac{x}{(x+y)} * 100 = \frac{0.12}{(0.12+0.027)} * 100$

$= 81.6\%$

% car not stolen $= \frac{y}{(x+y)} * 100 = \frac{0.027}{(0.12+0.027)} * 100 = 18.4\%$

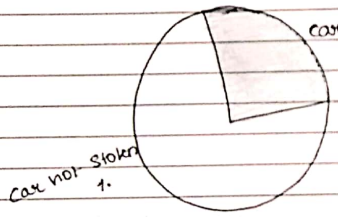
Write a program to implement the naive Bayesian classifier for a sample training data set stored as a .csv file. Compute the accuracy of the classifier, considering a few test data sets.

```
import csv
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
def Bayes(Data, x, col, yescount, nocount):
    xyes = 0
    xno = 0
    for line in Data:
        if line[col] == x:
            if line[-1] == 'Yes':
                xyes += 1
            else:
                xno += 1
    pxyes = xyes / yescount
    Pxno = xno / nocount
    return pxyes, Pxno
def main():
    file = "C:\\Users\\AMC College\\Desktop\\DATASET\\Car(2).csv
    temp = Pd.read_csv(file)
    display(temp)
    data = []
    fd = csv.reader(open(file))
    for line in fd:
        data.append(line);
    data = data[1:]
```

Output

	color	type	origin	Stolen
0	red	Sports	domestic	yes
1	red	Sports	Imported	no
2	Yellow	SUV	Imported	yes
3	red	Sports	domestic	yes
4	red	Sports	Imported	no
5	yellow	SUV	Imported	yes
6	yellow	SUV	Imported	yes
7	yellow	Sports	Imported	no
8	red	Sports	domestic	no
9	red	Sports	Imported	no

Enter your new car feature color, type, origin
red Sports domestic



Percentageyes = 28.571428571428577
percentageno = 71.42857142857143


```
n = len(data)
yescount = 0
nocount = 0
for line in Data:
  if line[-1] == 'yes':
    yes count + = 1
  else:
    nocount + = 1
pyes = yescount/n
pno = nocount/n
print("enter your new car feature color, type, origin")
x,y,z = input().split()
pxyes, pxno = Bayes(Data,x,0, yescount, nocount)
pyyes, pyno = Bayes(Data, y, 1, yescount, nocount)
p2yes, p2no = Bayes(Data, z, 2, yescount, nocount)
resyes = pxyes * pyyes * p2yes * pyes
resno = pxno * pyno * p2no * pno
percentageyes = resyes / (resyes + resno) * 100
percentageyes = resno / (resno + resyes) * 100
pex = [percentageyes, percentageno]
label = ["car stolen%", "car not stolen%"]
plt.pie(pex, labels = label)
plt.show()
main()
```

~~Output~~ Calculation

Sunny	Warm	Normal	Strong	warm	Same	yes
Sunny	warm	High	Strong	warm	Same	yes
Rainy	cold	High	Strong	warm	Change	No
Sunny	Warm	High	Strong	Cool	change	Yes

$S_0 = [\phi, \phi, \phi, \phi, \phi, \phi]$

$S_1 = [$Sunny, Warm, normal, Strong, warm, Same$]$

$S_2 = [$sunny, warm, ~~normal~~, ?, Strong, warm, Same$]$

$S_3 = [$Sunny, warm, ?, Strong, ?, ?$]$

Generalization

$G = [\langle$Sunny, ?, ?, ?, ?, ?$\rangle, \langle$?, warm, ?, ?, ?, ?$\rangle]$

For a given set of training data examples stored in a .csv file, implement and demonstrate the candidate elimination algorithm to output a description the set of all hypothesis consistent with training examples.

```
import csv
hypo = [ ]
gen = [ ]
datap = [ ]
dataN = [ ]
file = "C:\\Desktop\\AMCEC\\Swimming.csv"
fd = csv.reader (open(file))
for line in fd:
    print (line)
    if line[-1] == "Yes":
        datap.append (line)
    if line[-1] == "No":
        dataN.append(line)
print("---- positive example ----")
for line in datap:
    print (line)
print("----Negative example ----")
for line in dataN:
    print(line)
row= len(datap)
col = len(datap[0])
for j in range(col):
    hypo.append( data p[0][j])
for i in range (row):
    for j in range (col):
```

```
            if hypo[j] != data_p[i][j]:
                hypo[j] = ?
print("- - - - - Hypotheses - - - -")
print(hypo)
row = len(dataN)
col = len(dataN[0])
for i in range(row):
    for j in range(col-1):
        gen = [?, ?, ?, ?, ?, ?]
        if hypo[j] != dataN[i][j]:
            gen[j] = hypo[j]
            print(gen)
```

Output

['Sunny','warm','normal','Strong','warm','Same','yes']
['Sunny','warm','high','Strong','Warm','Same','yes']
['Rainy','cold','high','Strong','warm','Change','NO']
['Sunny','warm','high','Strong','Cool','Change','yes']

----Positive example-----
['Sunny','warm','normal','Strong','warm','Same','Yes']
['sunny','warm','high','Strong','warm','Same','Yes']
['Sunny','warm','high','Strong','Cool','Change','Yes']

-----Negative example-----
['Rainy','cold','high','Strong','warm','Change','NO']

------Hypothesis----
['Sunny','warm','?','Strong','?','?','Yes','Sunny','warm','Normal','Strong','warm','Same','Yes','Sunny','warm','Normal','Strong','warm','Same','Yes']
['Sunny','?','?','?','?','?']
['?','warm','?','?','?','?']
['?','?','?','?','?','?']
['?','?','?','?','?','?']
['?','?','?','?','?','?']