Report For The Assignment 1.

Pattern recognition & Machine Learning.

EE 657

IIT Guwahati.

Name: Pratik Ranjan.

Roll No.: 150122025.

Problem 1.

In this question our aim is to learn the Gaussian parameter to classify two different classes which are two numbers 5 and 6. We have two types of data sets one is training set and another is test set. From training set we will learn Gaussian parameters (mean and covariance) and from the learned parameters model we will learn the label of test data. Now we compare the learn label to the label of test data and form confusion matrix and calculate accuracy and misclassification rate.

Idea

We will use maximum likelihood approach to learn the parameter mean (μ) and covariance (∑) .

Since probability of class (Ci) given X can be given by equation:

$p(ci/x) = p(x/ci) * p(ci)/p(x)$ ----------(1)

Where p(x)=p(x/c5)p(c5)+p(x/c6)p(c6).(law of total probability). ----------(2)

Here probability of class give

n x p(ci/x) will be maximum if p(x/ci)(likelihood) will be maximum.

From Gaussian probability distribution we can define p(x/ci) as follow:

$p(x/ci) \sim N(x/\mu i, \sum i)$

$N(x/\mu i, \sum i) = \frac{1}{(2\pi)^{\frac{d}{2}} * \sqrt{|\sum i|}} * e(-\frac{1}{2}(x-ui)^T \sum i^{-1}(x-ui))$

d= dimension of the data (attribute here d=64).

|∑i|=determinant of the covariance matrix.

$(x-\mu)\sum^{-1}(x-\mu)^T$= Mahalanobis Distance.

From maximum likelihood we know:-

$\mu i = \frac{\sum_j^N x*r^i}{\sum_j^N r^i}$ where ri is label.

$\sum i = \frac{(x-ui)^T(x-ui)}{Ni}$

Where xi is the data set belonging to class i and Ni is the number of data set belonging to the class i.

Here I am calculating mean and covariance for separate class and counting number of training sets of class 5 (c5) and class 6 (c6).

```
"""--------------------calculate mean and varience-------------------------"""
mu5=0
mu6=0
c5=0
c6=0
cov5=0
cov6=0
x5=[]
x6=[]
for i in range(0,rt):
    if trlabel[i]==5:
        mu5=mu5+trdata[i]
        x5.append(trdata[i])
        c5=c5+1
    else:
        mu6=mu6+trdata[i]
        x6.append(trdata[i])
        c6=c6+1


x5=np.matrix(x5)
x6=np.matrix(x6)

mu5=mu5/c5
mu5=np.matrix(mu5)
cov5=np.matmul(np.transpose(x5-mu5),(x5-mu5))
cov5=cov5/c5;

mu6=mu6/c6
mu6=np.matrix(mu6)
cov6=np.matmul(np.transpose(x6-mu6),(x6-mu6))
cov6=cov6/c6;
```

After learning mean and covariance of each class I have calculated likelihood probability or p(x/ci) (here x is test data set) by using Gaussian distribution as follow:-

```
"""--------------calculating gaussian-----------------------------"""
prc5=c5/rt;
prc6=c6/rt;


rts=len(tsdata)
cts=len(tsdata[0])

cov5inv=np.linalg.inv(cov5)
cov5det=np.linalg.det(cov5)
cov5det=m.sqrt(abs(cov5det))
p5=(m.sqrt(2*m.pi))**cts
d5=p5*cov5det
#print("\n")
cov6inv=np.linalg.inv(cov6)
cov6det=np.linalg.det(cov6)
cov6det=m.sqrt(abs(cov6det))
p6=(m.sqrt(2*m.pi))**cts
d6=p6*cov6det
lrnlabel=[]
for i in range(0,rts):
    #calculate pxgc5
    x=(np.matrix(tsdata[i])-mu5)
    pr=np.matmul(x,np.transpose(cov5inv))
    z=np.matmul(pr,np.transpose(x))
    pxgc5=(np.exp(-z/2))*prc5
    pxgc5=pxgc5/(d5)
    #calculate pxgc6
    x=(np.matrix(tsdata[i])-mu6)
    pr=np.matmul(x,np.transpose(cov6inv))
    z=np.matmul(pr,np.transpose(x))
    pxgc6=(np.exp(-z/2))*prc6
    pxgc6=pxgc6/(d6)
    #calculate pc5gx
    pc5gx=pxgc5/(pxgc5+pxgc6)
    #learning label
    if(pc5gx>0.5):
        lrnlabel.append(5)
    else:
        lrnlabel.append(6)
```

Prc5 is probability of class 5= (number of training set belonging to class 5)/(total number of training).

Similarly prc6 has been calculated.

Now after calculating pxgc5 and pxgc6 (which probability of x given class i) we can calucalate p(x/ci)

Here I have calculated pc5gx (p(c5/x)) by using conditional probability and total law of probability (eqn 1 and 2). If p(c5/x)>0.5 then I have assigned label 5 to the data else assigned label 6 to the data (labelling is stored in lrnlabel).

After learning the label I have calculated the confusion matrix as follow:-

```
"""------------calcuating misclassification-------------------------"""
misc=[[0,0],[0,0]]
for i in range(0,rts):
    if(lrnlabel[i]==5 and tslabel[i]==5):
        misc[0][0]=misc[0][0]+1
    elif(lrnlabel[i]==5 and tslabel[i]==6):
        misc[0][1]=misc[0][1]+1
    elif(lrnlabel[i]==6 and tslabel[i]==5):
        misc[1][0]=misc[1][0]+1
    else:
        misc[1][1]=misc[1][1]+1

ct5=0
ct6=0
for i in range(0,rts):
    if(tslabel[i]==5):
        ct5=ct5+1
    else:
        ct6=ct6+1
print("confusion matrix:-")
print(misc)
misc[0][0]=(misc[0][0]/ct5)*100
misc[0][1]=(misc[0][1]/ct6)*100
misc[1][0]=(misc[1][0]/ct5)*100
misc[1][1]=(misc[1][1]/ct6)*100
print("percentage:-")
print(misc)
```

Here I am comparing label from learned label and test label and storing number of match and mismatch in confusion matrix.

In confusion matrix (misc):-

misc[0][0] is number of matches of 5 in learned label to test label (positive-positive match)

misc[0][1] is number of misclassification of learned label 5 to label 6 (positive –negative mismatch)

misc[1][0] is number of misclassification of learned label 6 to label 5 (negative –positive mismatch)

misc[1][1] is number of matches of 6 in learned label to test label (negative -negative match)

after calculating confusion matrix I have calculated accuracy and misclassification rate by calculating percentage of each element of matrix according to number of test set for each class.

Results :-

(1) If we use calculated covariance for 5 and 6 we will get:

Confusion matrix = $\begin{matrix} 106 & 27 \\ 49 & 151 \end{matrix}$

Misclassification (rate matrix)= $\begin{matrix} 68.38 & 15.16 \\ 31.61 & 84.83 \end{matrix}$

(2) If cov5=cov6:

Confusion matrix = $\begin{matrix} 110 & 11 \\ 45 & 167 \end{matrix}$

Misclassification (rate matrix) = $\begin{matrix} 70.96 & 6.1 \\ 29.03 & 93.82 \end{matrix}$

(3) If cov6=cov5:

Confusion matrix = $\begin{matrix} 139 & 50 \\ 16 & 128 \end{matrix}$

Misclassification (rate matrix) = $\begin{matrix} 89.67 & 28.09 \\ 10.32 & 71.91 \end{matrix}$

Conclusion :-

If we are taking covariance matrix of class 6 then we can see that accuracy of both class 5 and class 6 are increased so from result it is better to use covariance 6 for both classes.

Problem 2.

This problem is extension of the problem 1. Here we have to plot iso-contour plot and and discriminant function for the model using different variations of covariance.

Idea.

Idea is same as first problem.

Only I am adding new code for plotting contour plot as follow:-

```
"""------------------------Ploting Contour--------------------------"""
#ploting contour
import matplotlib.cm as cm
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt

matplotlib.rcParams['xtick.direction'] = 'out'
matplotlib.rcParams['ytick.direction'] = 'out'


cov0=np.array(cov0)
cov1=np.array(cov1)

mu0=np.array(mu0)
mu1=np.array(mu1)
"""--------function for gaussian calculation------------"""
def bivariate_normal(position, mu, cov):
    n = mu.shape[0]
    cov_det = np.linalg.det(cov)
    cov_inv = np.linalg.inv(cov)
    d = np.sqrt((2*np.pi)**n * cov_det)
    fac = np.einsum('...k,kl,...l->...', position-mu, cov_inv, position-mu)
    return np.exp(-fac / 2) / d

"""----------function end--------------------------"""
```

```
"""-----------function end-----------------------------"""
sd=np.linspace(min(tsdata[:,0]),max(tsdata[:,0]),1000)
sd1=np.linspace(min(tsdata[:,1]),max(tsdata[:,1]),1000)
X, Y = np.meshgrid(sd,sd1)
dis = np.empty(X.shape + (2,))
dis[:, :, 0] = X
dis[:, :, 1] = Y


Z0 = bivariate_normal(dis,mu0,cov0)
plt.contour(X,Y,Z0)

Z1 = bivariate_normal(dis,mu1,cov1)
plt.contour(X,Y,Z1)

# difference of Gaussians
Z = (Z1 - Z0)
plt.contour(X,Y,Z)

plt.title('Iso-probability Contour')
plt.scatter(tsdata[:,0],tsdata[:,1])
plt.ylabel('x2')
plt.xlabel('x1')
plt.show()
```

Here I made bivariate_normal function which is calculating bivariate normal distribution.

Since we have to plot contour so, I formed X Y meshgrid for 2D plot.

X Y meshgrid are made from the continuous point. The continuous points are made from minimum and maximum point of test data. This is done to avoid distorted plot.

After that I initialised distance in three dimensional (dis) matrix.

Now I passed appropriate parameter to the function and calculated contour points and plotted them on the same figure for both classes.

To plot discriminant function I subtracted contour of class 0 from class 1

Z = Z0-Z1

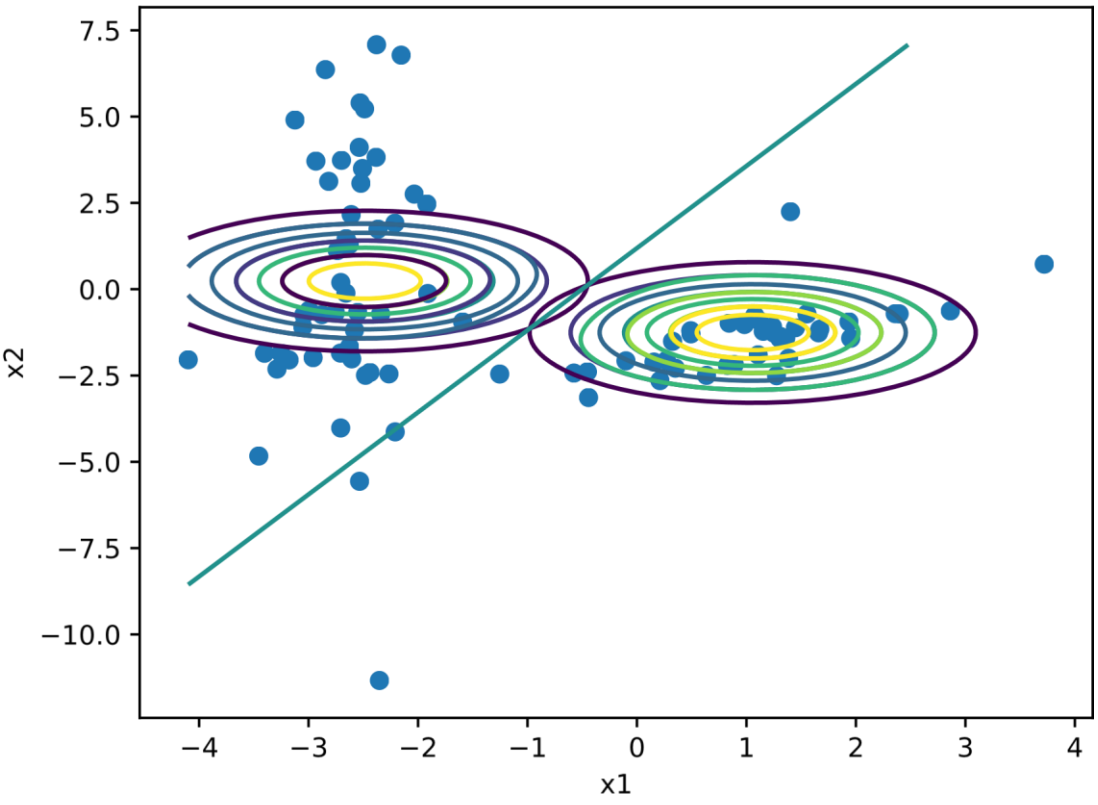
Z is discriminant function. Now I plotted Z on the same figure.

Result.

Case1.

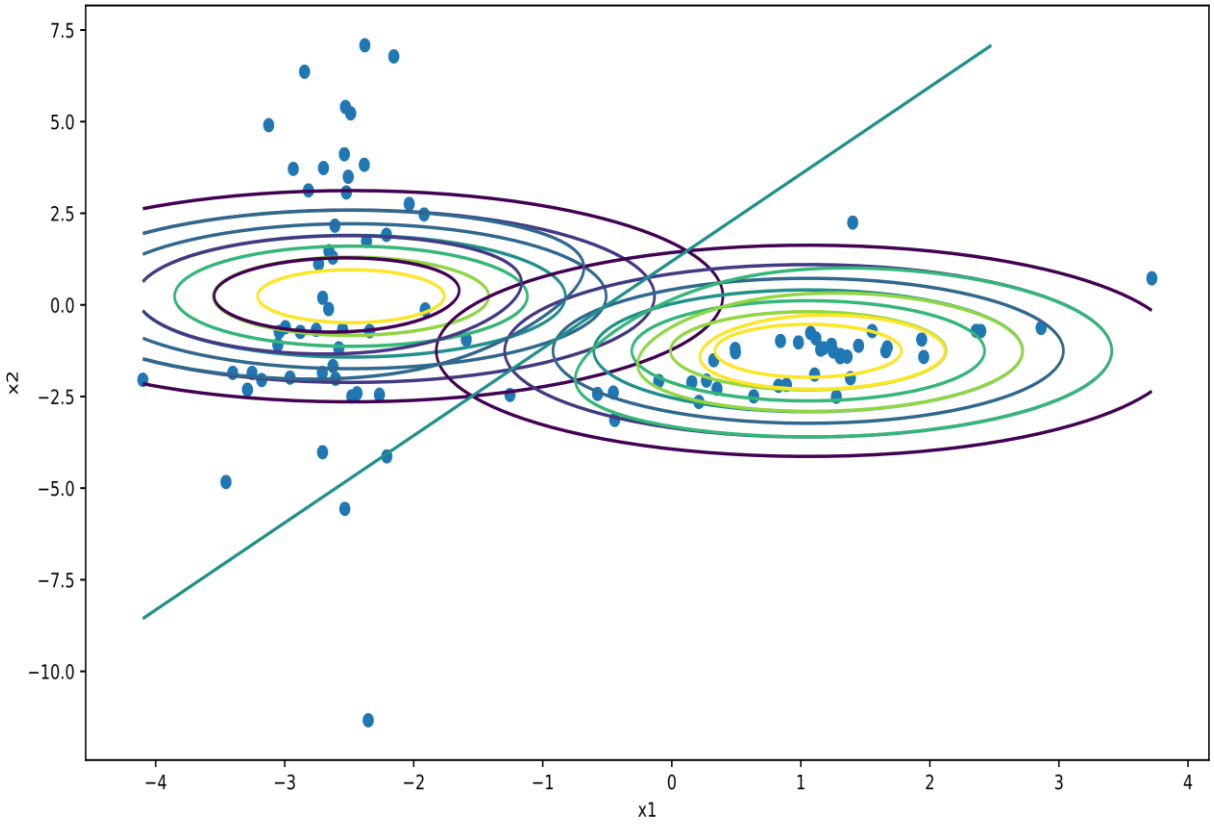
Cov0=$\begin{matrix} a & 0 \\ 0 & a \end{matrix}$ and cov1=cov0.

(1) If a = 1. (isoprobability contour at a=1)

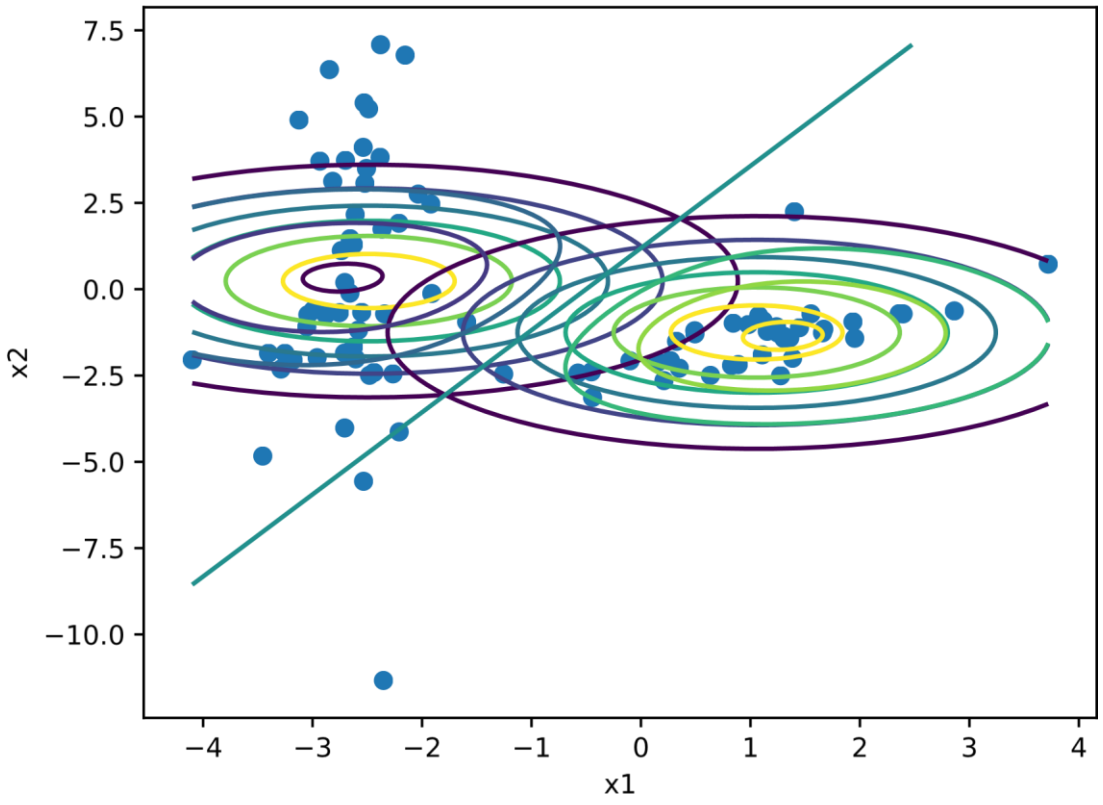


Green straight line is decision boundary.

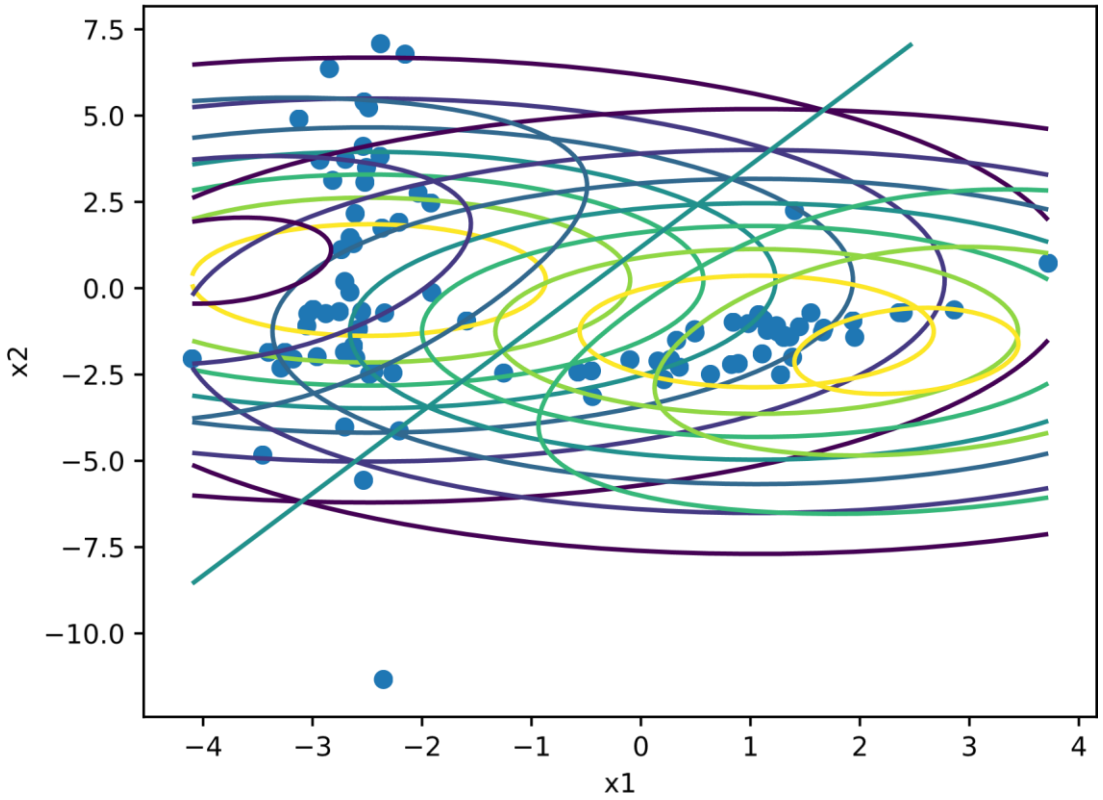
(2) If a = 2 (isoprobability contour at a=2)



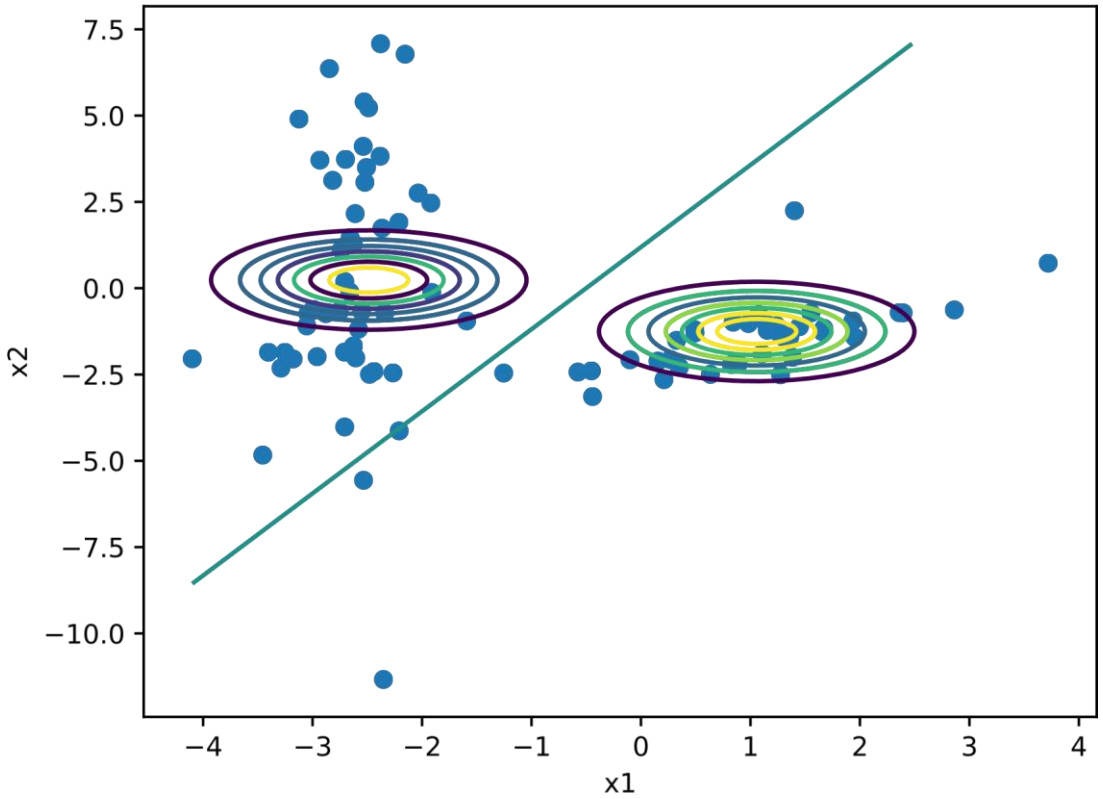
(3) If a =3 (isoprobability contour at a=3)



(4) If a = 10 (isoprobability contour at a=10)



(5) If a = 0.5 (isoprobability contour at a = 0.5)



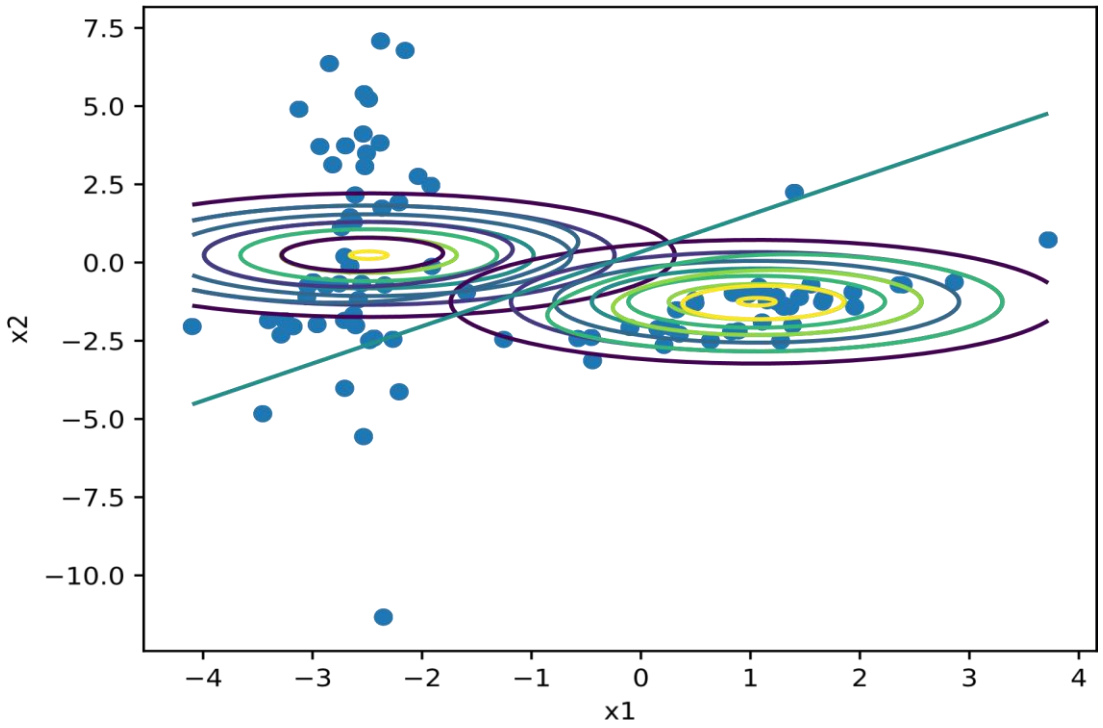
For all different value of a confusion matrix and misclassification rate are same.

Confusion matrix = $\begin{matrix} 47 & 0 \\ 3 & 40 \end{matrix}$ misclassification rate = $\begin{matrix} 94 & 0 \\ 6 & 100 \end{matrix}$

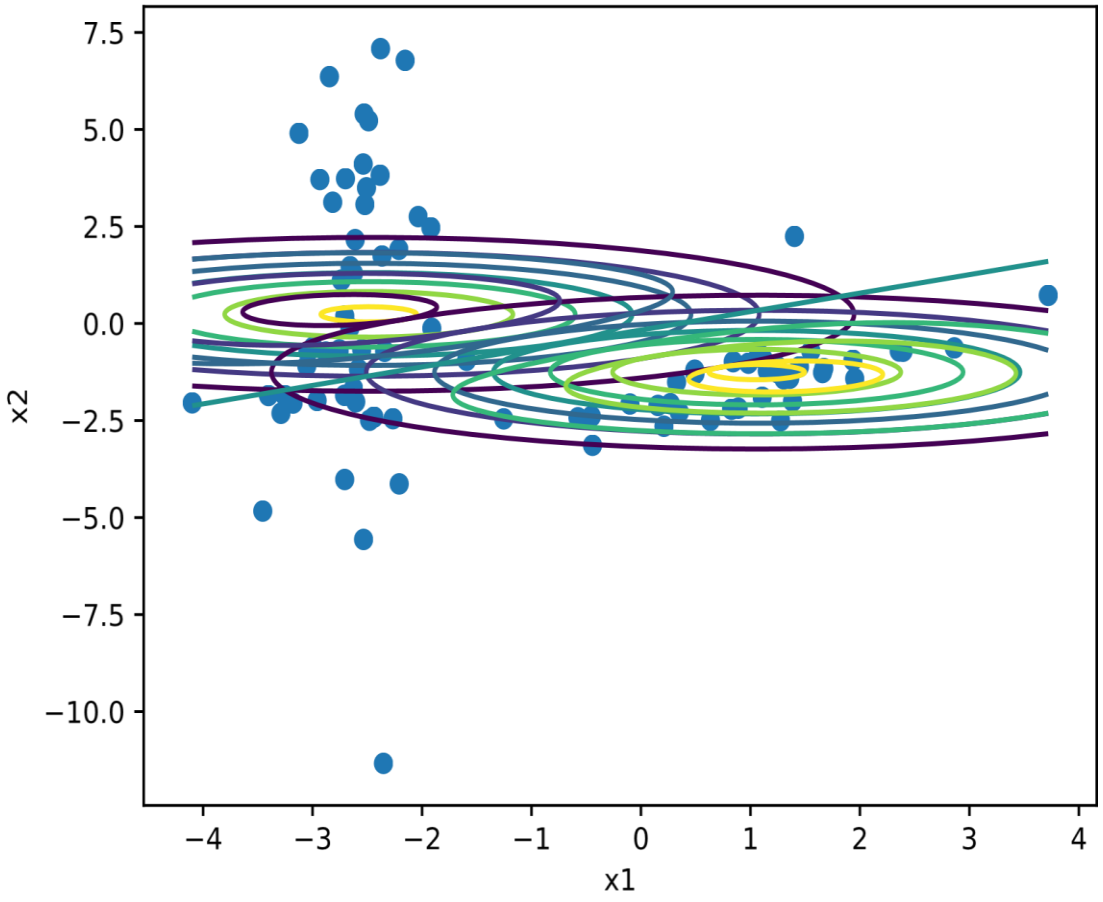
Case2.

Cov1 = $\begin{matrix} a & 0 \\ 0 & b \end{matrix}$ and cov0 = cov1.

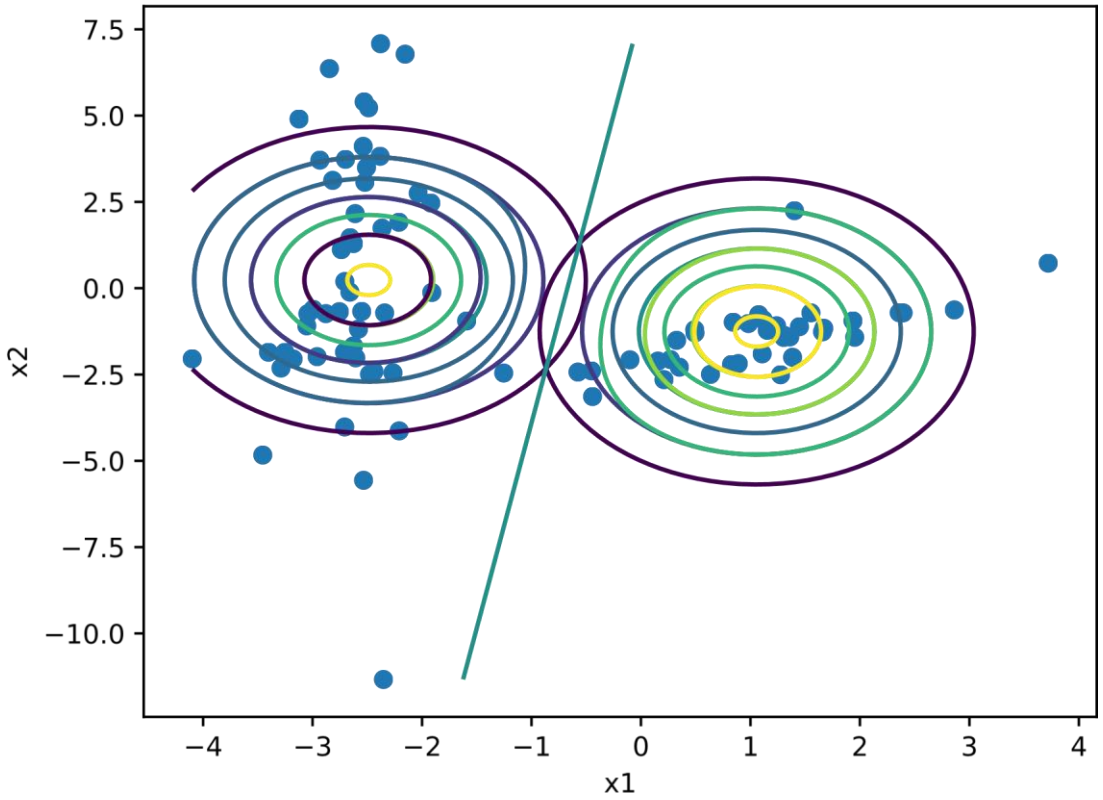
(1) If a = 2 and b = 1



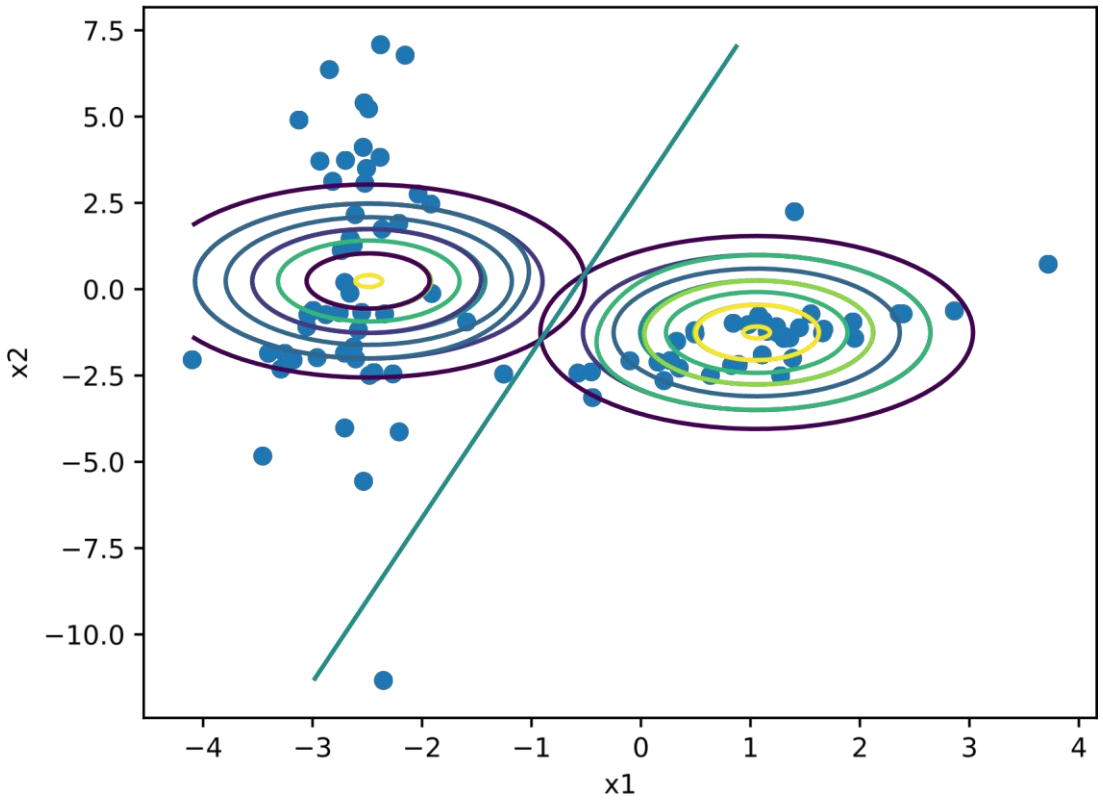
(2) If a = 5 and b = 1



(3) a = 1 and b = 5



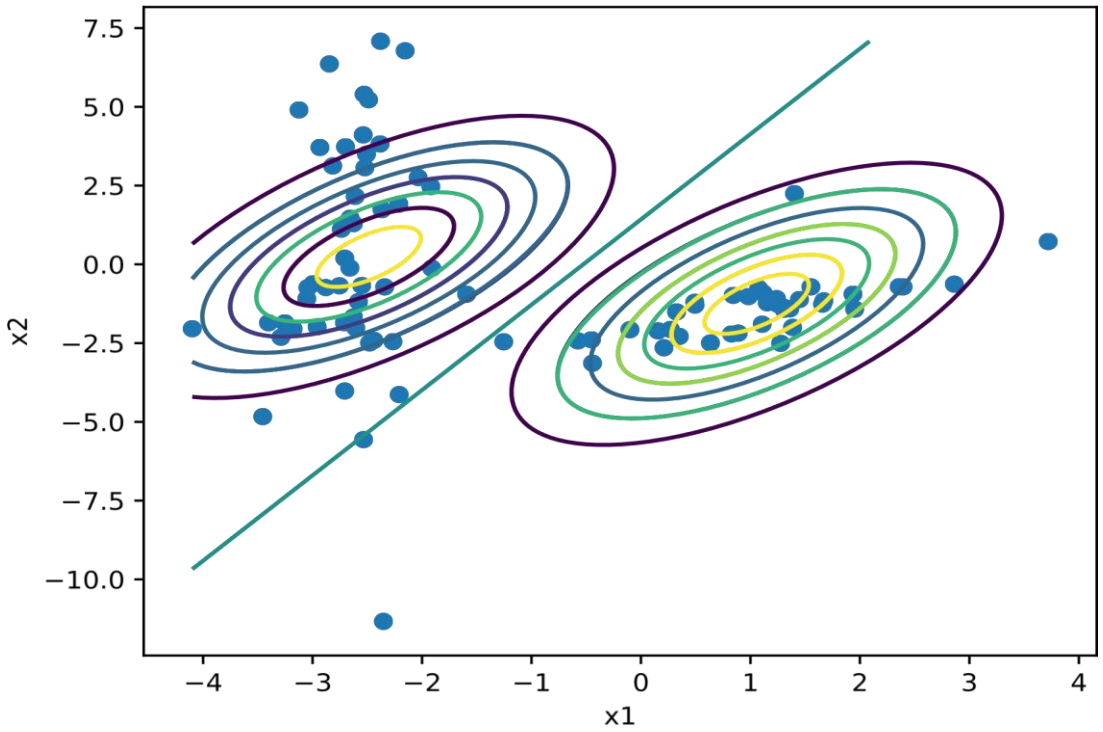
(4) a = 1 and b = 2



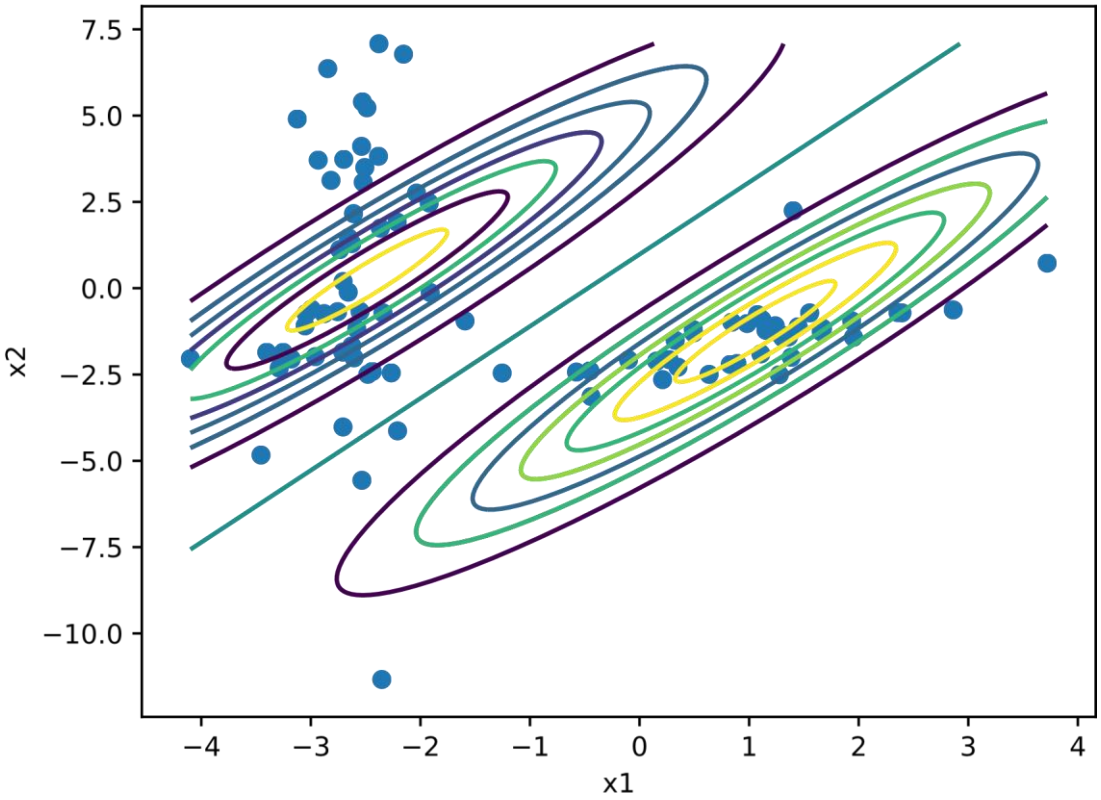
Case3.

Cov1 = $\begin{matrix} a & b \\ c & d \end{matrix}$ and cov0 = cov1.

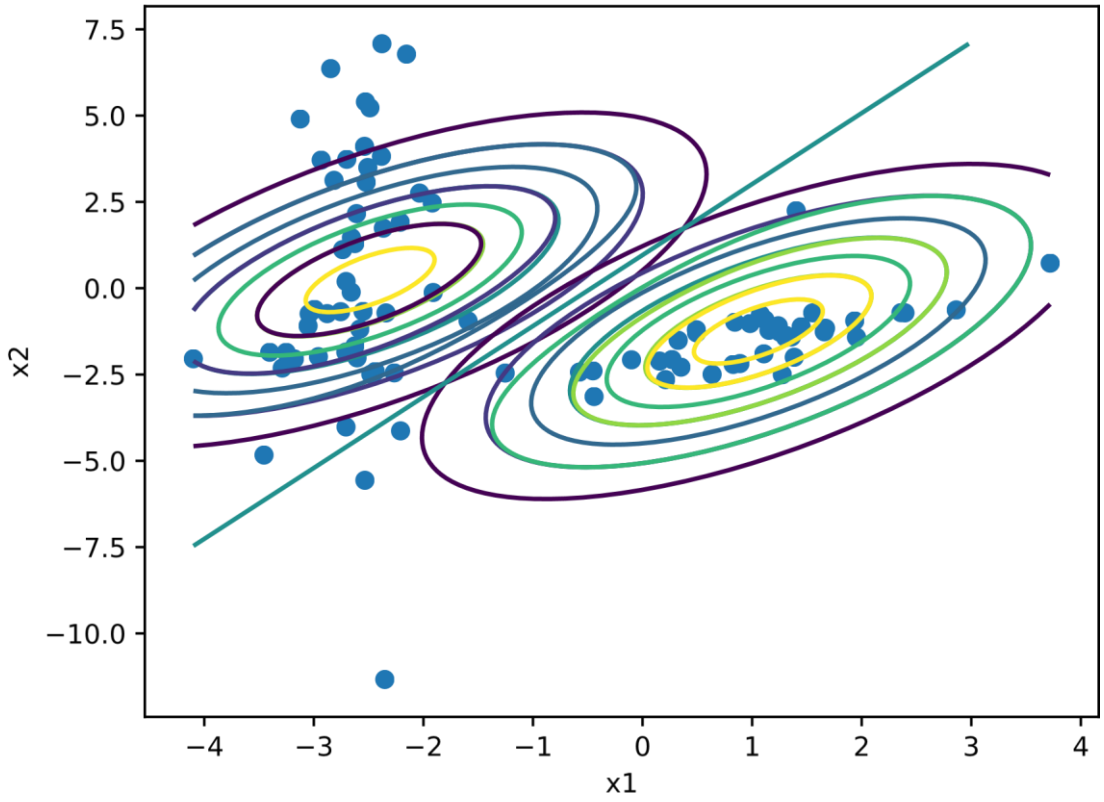
(1) If a=1,b=0.5,c=2,d=4



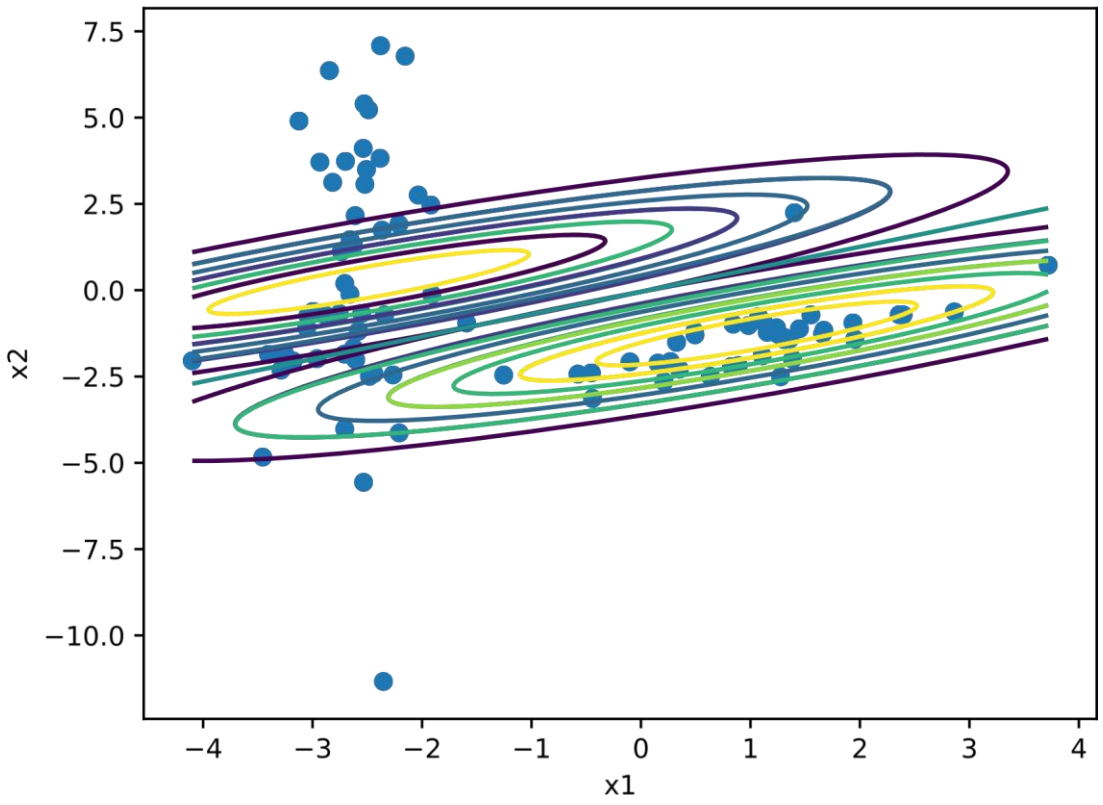
(2) If a=1,b=0.75,c=3,d=4



(3) If a=2,b=1,c=3,d=5



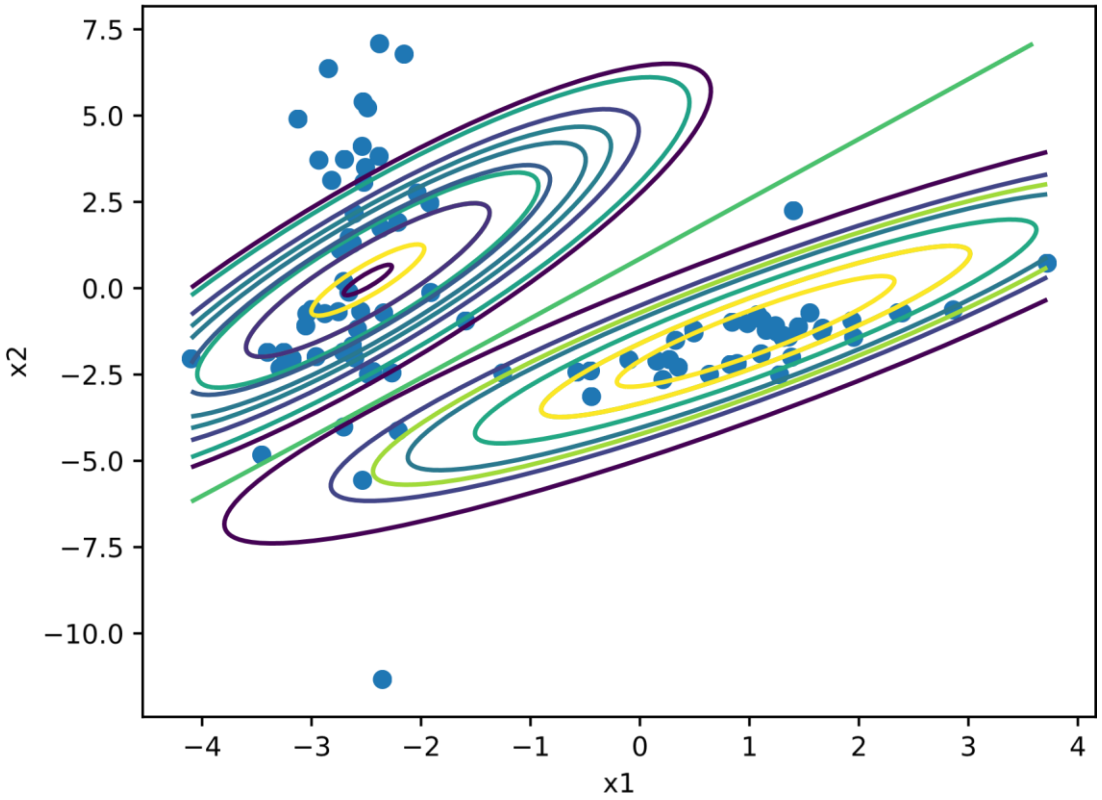
(4) If a=5,b=4,c=1.5,d=2



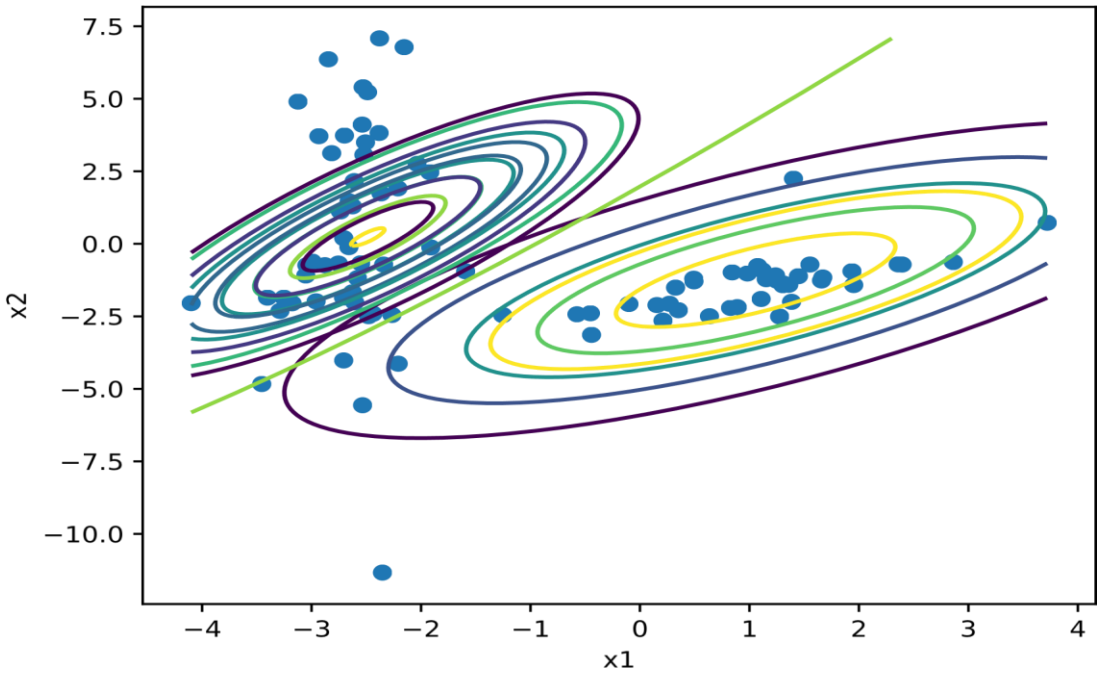
Case4.

Cov1 = $\begin{matrix} e & f \\ g & h \end{matrix}$ and Cov0 = $\begin{matrix} a & b \\ c & d \end{matrix}$

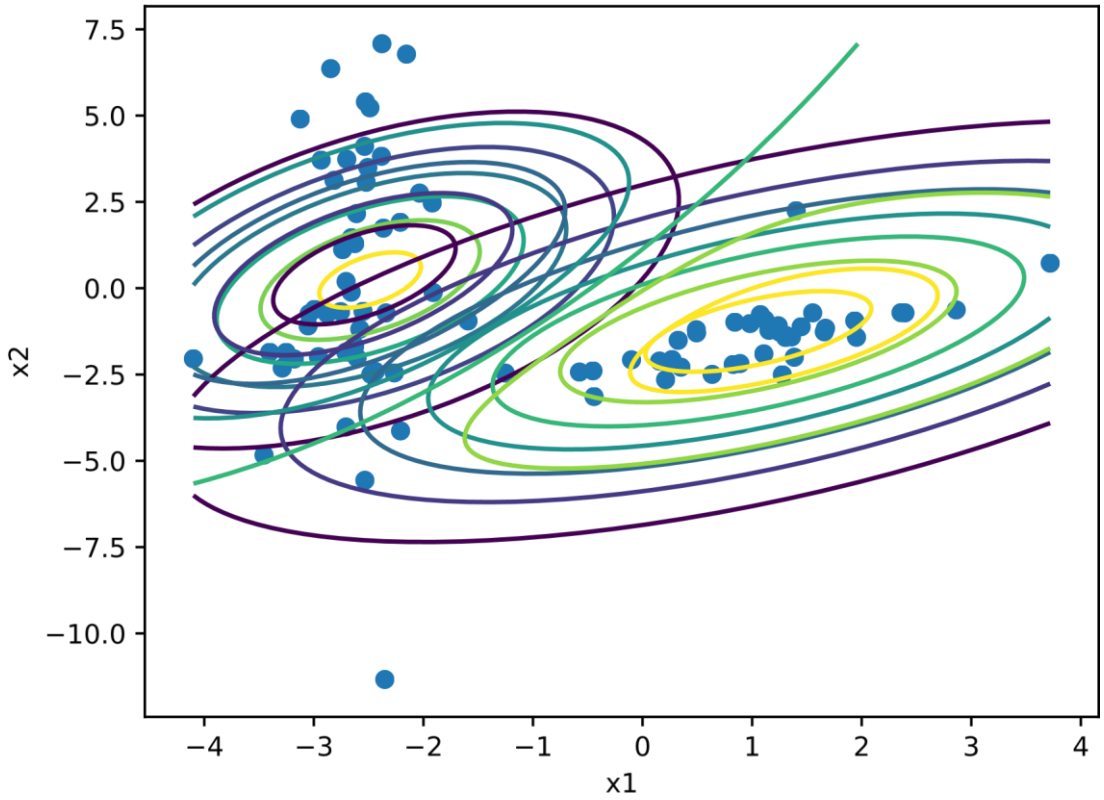
(1) If (a= 1,b= 0.5,c=3,d=4) (e=5,f=4.5,g=7,h=8)



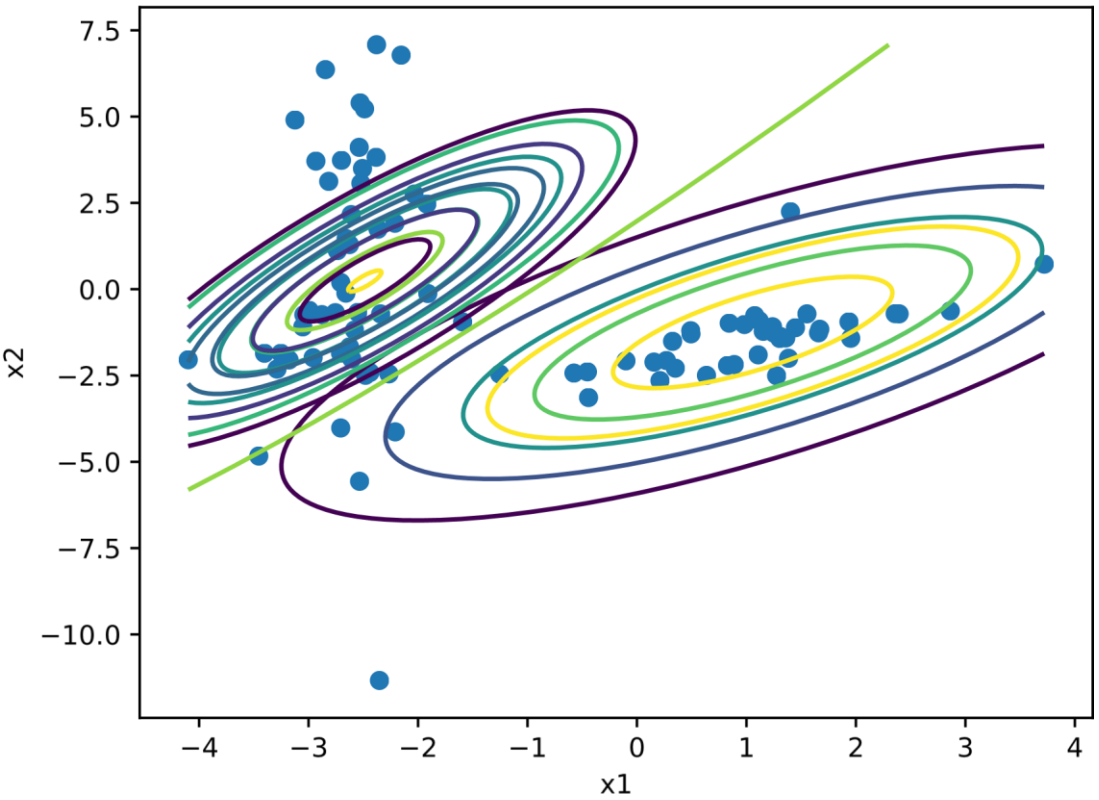
(1) (2) if (a= 1,b= 0.8,c=2.5,d=4) (e=5,f=3.5,g=5.5,h=8)



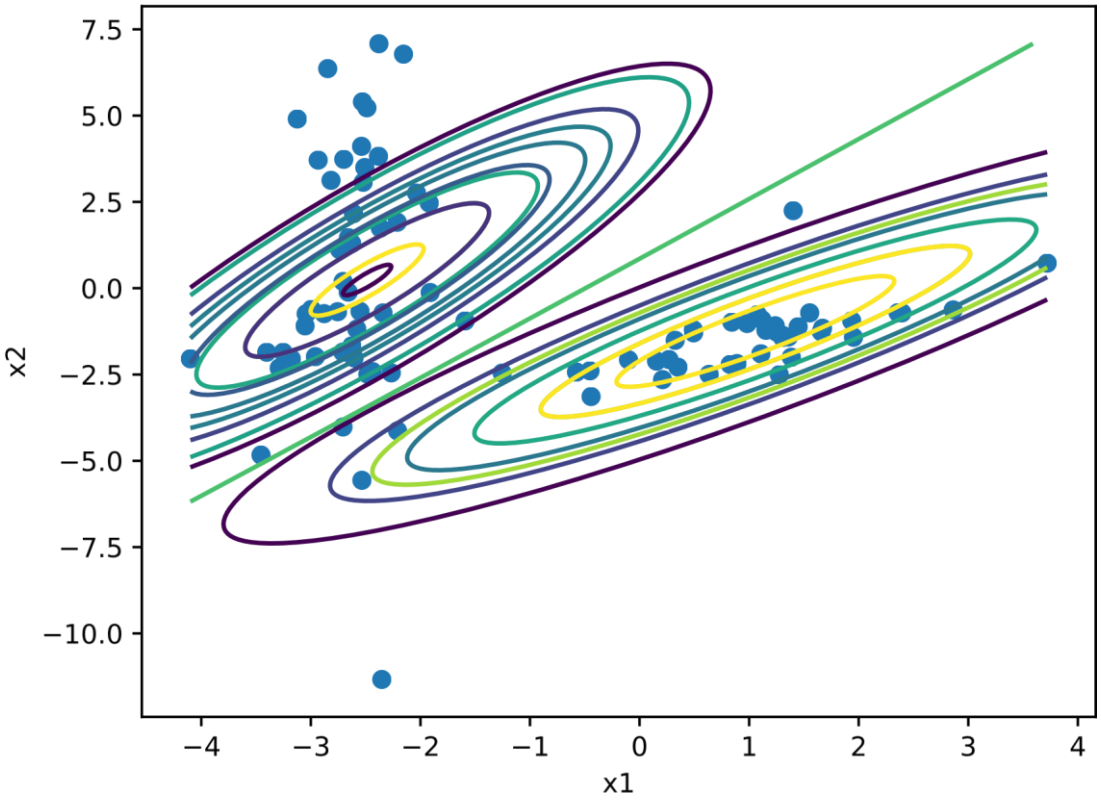
(1) (3) if (a= 2,b= 0.8,c=2.5,d=4) (e=7,f=3.5,g=5.5,h=9)



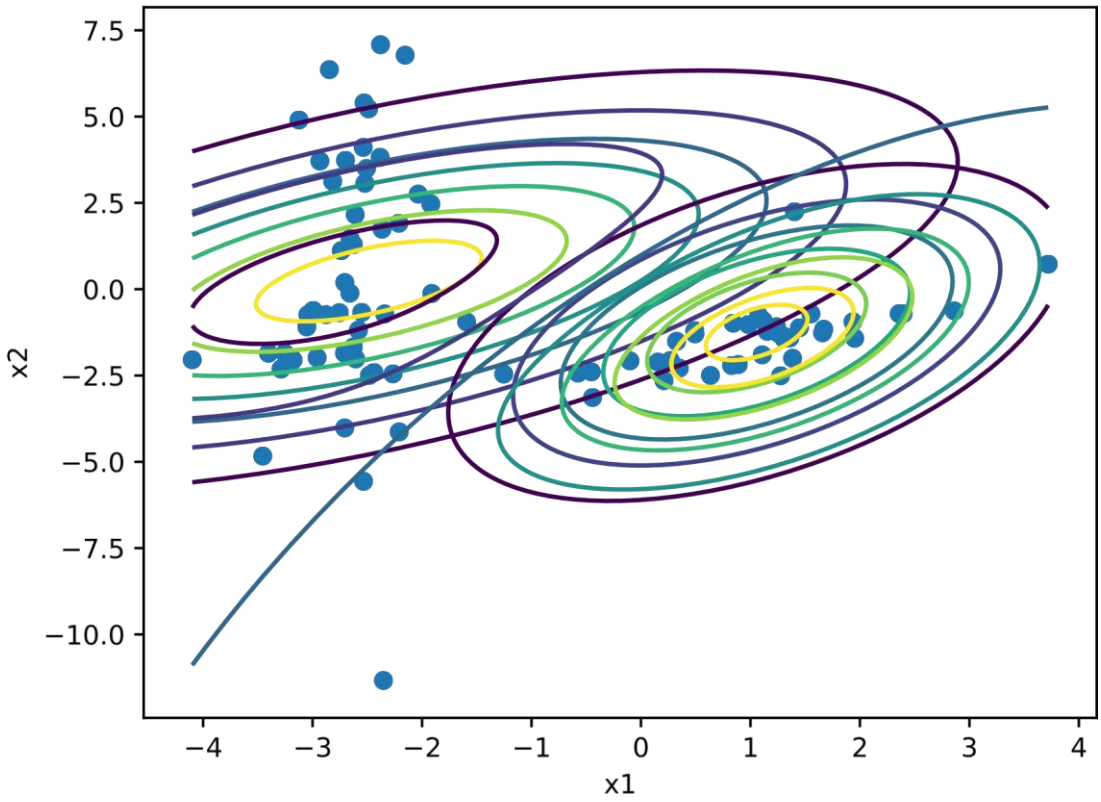
(4) if if (e= 2,f= 0.8,g=2.5,h=4) (a=7,b=3.5,c=5.5,d=9)



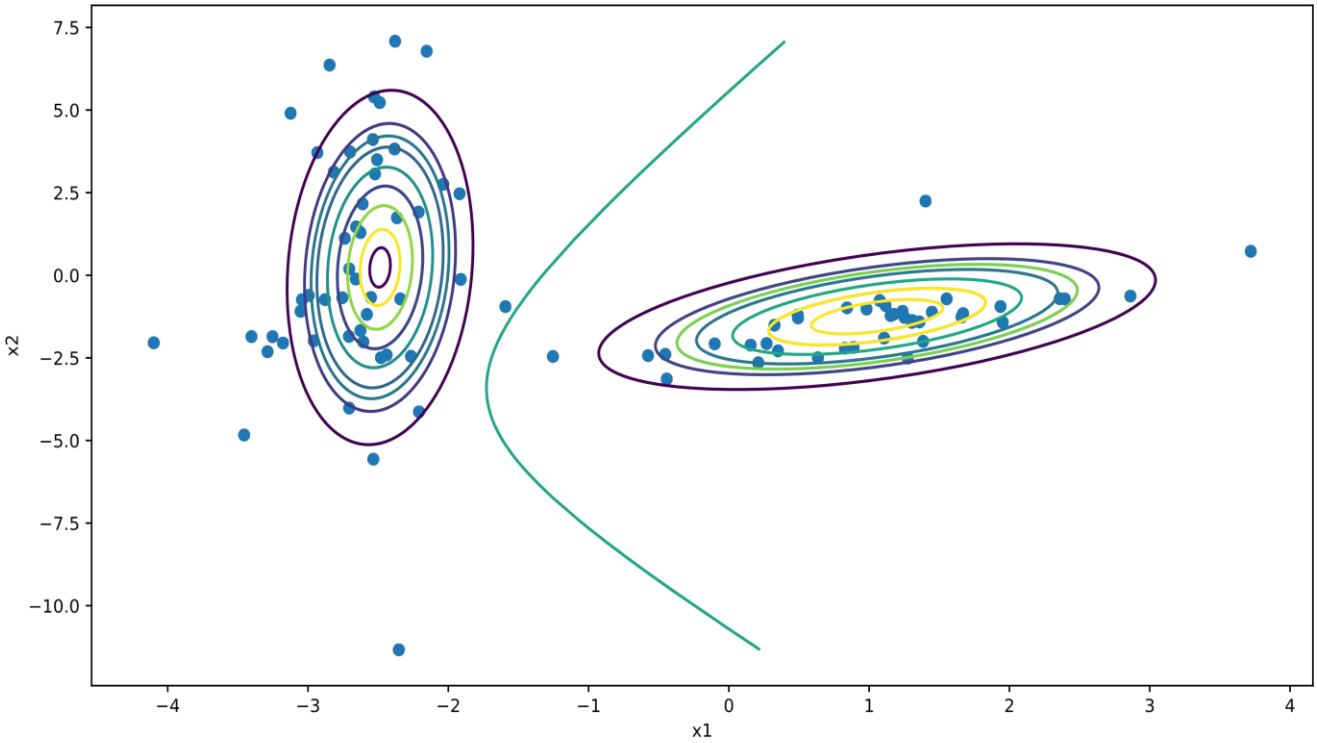
(5)


7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
x2
−4
−3
−2
−1
0
1
2
3
4
x1

(6)


7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
x2
−4
−3
−2
−1
0
1
2
3
4
x1

Plot for calculated covariance cov0 and cov1:-



Calculated covariance and confusion matrix:

Cov0 = $\begin{matrix} 0.107 & 0.106 \\ 0.106 & 7.04 \end{matrix}$ cov1= $\begin{matrix} 1.037 & 0.578 \\ 0.578 & 1.280 \end{matrix}$

Confusion matrix = $\begin{matrix} 50 & 0 \\ 0 & 40 \end{matrix}$ misclassification rate = $\begin{matrix} 100 & 0 \\ 0 & 100 \end{matrix}$

Conclusion:-

Case1:- by increasing variance value we can see from the plot that overlapping of contour of both plots are increasing and by decreasing the variance overlapping of both plots are decreasing. For all variations of a I observed confusion matrix remains sames.

Case 2:- by keeping value of “b” constant and increasing value of “a” we can see from the plot that elliptical overlapping is increasing and decision boundary is shifting towards X1.

by keeping value of “a” constant and increasing value of “b” we can see from the plot that circular overlapping is increasing and decision boundary is shifting towards X2.

Case3:-

I observed that covariance should be less than variance.

That is sigma(ij)<sigma(i^2).

By increasing covariance(sigma ij) elliptical contour is increasing. But decision boundary is not changing.

Case4:-

I have drawn plot arbitrary values and also for different covariance then I observed that these contours are combination of above 3 cases.

Contour drawn from the learned covariance is better as decision boundary drawn in this plot is clearly classifying both classes.

Problem3.

In this problem we have to make three model for predicting Wage by using linear regression method. We have to make models as a function of Age, Year and Education. Data set is given in the question.

Idea.

In linear regression we fit parameters and degree of polynomial. I have done It by using list square fit method. In least square fit method we minimizes the square error.

To find parameters (W) we make a matrix (A) which include sum of power of variable up to degree of polynomial . we are taking column matrix Y which include sum of product of output value and given variable to the power increasing up to degree of polynomial.

A*W=Y

Therefor W= inverse(A)*Y.

We are basically solving linear equations which contains n parameters and n equations where n is degree of polynomial.

A , W and Y can be calculated in this way.

$$\mathbf{A} = \begin{bmatrix} N & \sum_t x^t & \sum_t (x^t)^2 & \cdots & \sum_t (x^t)^k \\ \sum_t x^t & \sum_t (x^t)^2 & \sum_t (x^t)^3 & \cdots & \sum_t (x^t)^{k+1} \\ \vdots & & & & \\ \sum_t (x^t)^k & \sum_t (x^t)^{k+1} & \sum_t (x^t)^{k+2} & \cdots & \sum_t (x^t)^{2k} \end{bmatrix}$$

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_k \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} \sum_t r^t \\ \sum_t r^t x^t \\ \sum_t r^t (x^t)^2 \\ \vdots \\ \sum_t r^t (x^t)^k \end{bmatrix}$$

Here N is number of data set.

This is code for linear regression of wage vs age.

```
"""---------------------------------wage vs age linear regression-----------------------------"""
print("wage vs age:-")
print("Enter the degree of polynomial to be fit:")
dim=int(input())
#dim=10
dim=dim+1
yage=[]
a=np.zeros((dim,dim))
#print(a)
for i in range(0,dim):
    yage.append(np.sum(wage*(age**i)))
    for j in range(0,dim):
        a[i][j]=np.sum(age**(i+j))

#print(a)
agvwg_par=np.matmul(np.linalg.inv(np.matrix(a)),np.transpose(np.matrix(yage)))
#print(agvwg_par)

learnwage=[]

for i in range(0,nr):
    sm=0
    for j in range(0,dim):
        sm=sm+agvwg_par[j]*(age[i]**j)
        sm=float(sm)
    learnwage.append(sm)

learnwage=np.array(learnwage)
#print(learnage)
#print(age)
plt.title('Wage vs Age plot')
plt.ylabel('Wage')
plt.xlabel('Age')
plt.plot(age,wage,'bo',age,learnwage,'r^')
plt.show()
```

Here a is Matrix A.

Yage is Y Matrix.

In agvwg_par (W) I am calculating parameter for model wage as function of age.

I learnwage I am taking predicted value of wage from the corresponding age.

After that I am plotting Wage vs age plot.

This is code for linear regression of wage vs year.

```
"""--------------------------------wage vs year linear regression----------------------------
print("wage vs year:-")
print("Enter the degree of polynomial to be fit:")
dim=int(input())
#dim=10
dim=dim+1
yr=[]
a=np.zeros((dim,dim))

for i in range(0,dim):
    yr.append(np.sum(wage*(year**i)))
    for j in range(0,dim):
        a[i][j]=np.sum(year**(i+j))


yrvwg_par=np.matmul(np.linalg.inv(np.matrix(a)),np.transpose(np.matrix(yr)))

learnwage=[]

for i in range(0,nr):
    sm=0
    for j in range(0,dim):
        sm=sm+yrvwg_par[j]*(year[i]**j)
        sm=float(sm)
    learnwage.append(sm)

learnwage=np.array(learnwage)

plt.title('Wage vs year plot')
plt.ylabel('Wage')
plt.xlabel('year')
plt.plot(year,wage,'bo',year,learnwage,'r^')
plt.show()
```

Here variable is Year.

Here a is Matrix A.

yr is Y Matrix.

In yrvwg_par (W) I am calculating parameter for model wage as function of Year.

I learnwage I am taking predicted value of wage from the corresponding year.

After that I am plotting Wage vs year plot.

This is code for linear regression of wage vs education.

```
"""------------------------------wage vs education linear regression------------------------------"""
print("wage vs education:-")
print("Enter the degree of polynomial to be fit:")
dim=int(input())
dim=dim+1
yed=[]
a=np.zeros((dim,dim))

for i in range(0,dim):
    yed.append(np.sum(wage*(ed**i)))
    for j in range(0,dim):
        a[i][j]=np.sum(ed**(i+j))


edvwg_par=np.matmul(np.linalg.inv(np.matrix(a)),np.transpose(np.matrix(yed)))

learnwage=[]

for i in range(0,nr):
    sm=0
    for j in range(0,dim):
        sm=sm+edvwg_par[j]*(ed[i]**j)
        sm=float(sm)
    learnwage.append(sm)

learnwage=np.array(learnwage)

plt.title('Wage vs education plot')
plt.ylabel('Wage')
plt.xlabel('education')
plt.plot(ed,wage,'bo',ed,learnwage,'r^')
plt.show()
```

Here variable is Education.

Here a is Matrix A.

yed is Y Matrix.

In edvwg_par (W) I am calculating parameter for model wage as function of education.

I learnwage I am taking predicted value of wage from the corresponding education.

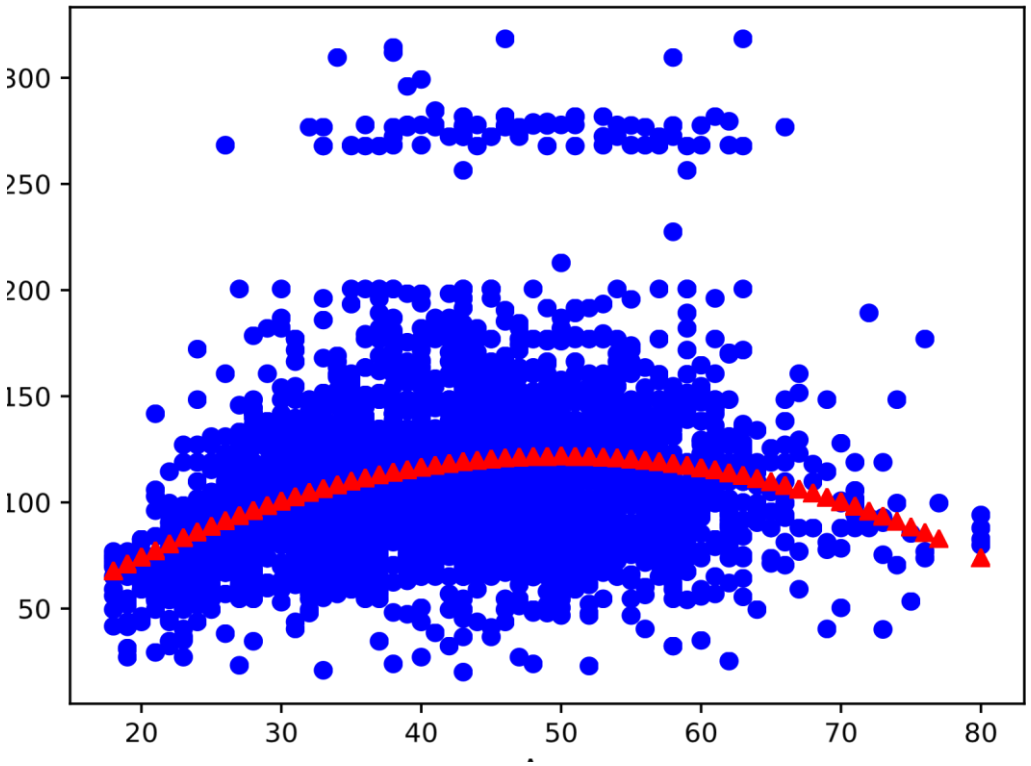
After that I am plotting Wage vs education plot.

Result:-

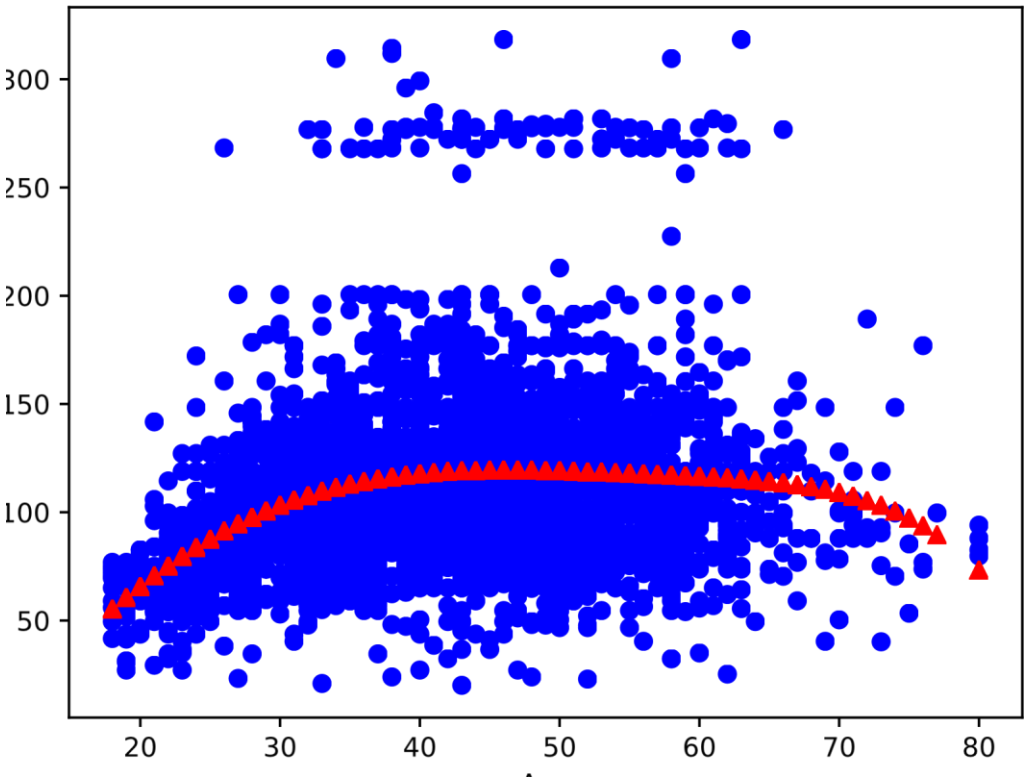
Note:-Red dots are fitted plot and blue dots are given data.

Wage vs Age plot(y = wage x = age).

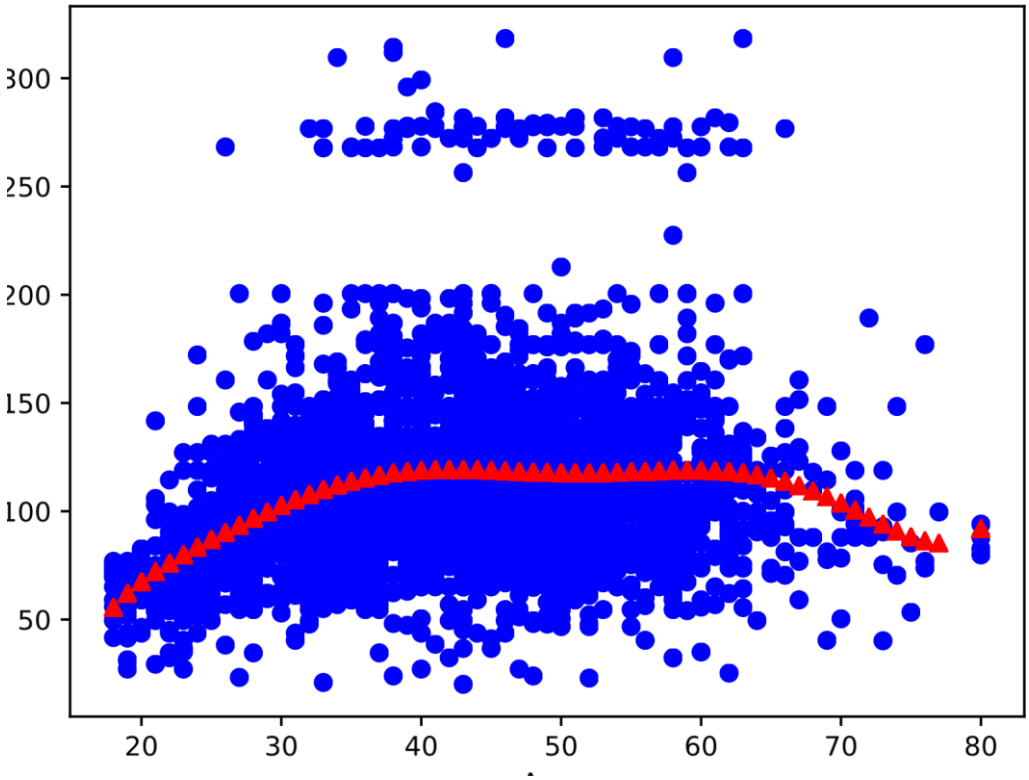
(1) Degree of polynomial = 2



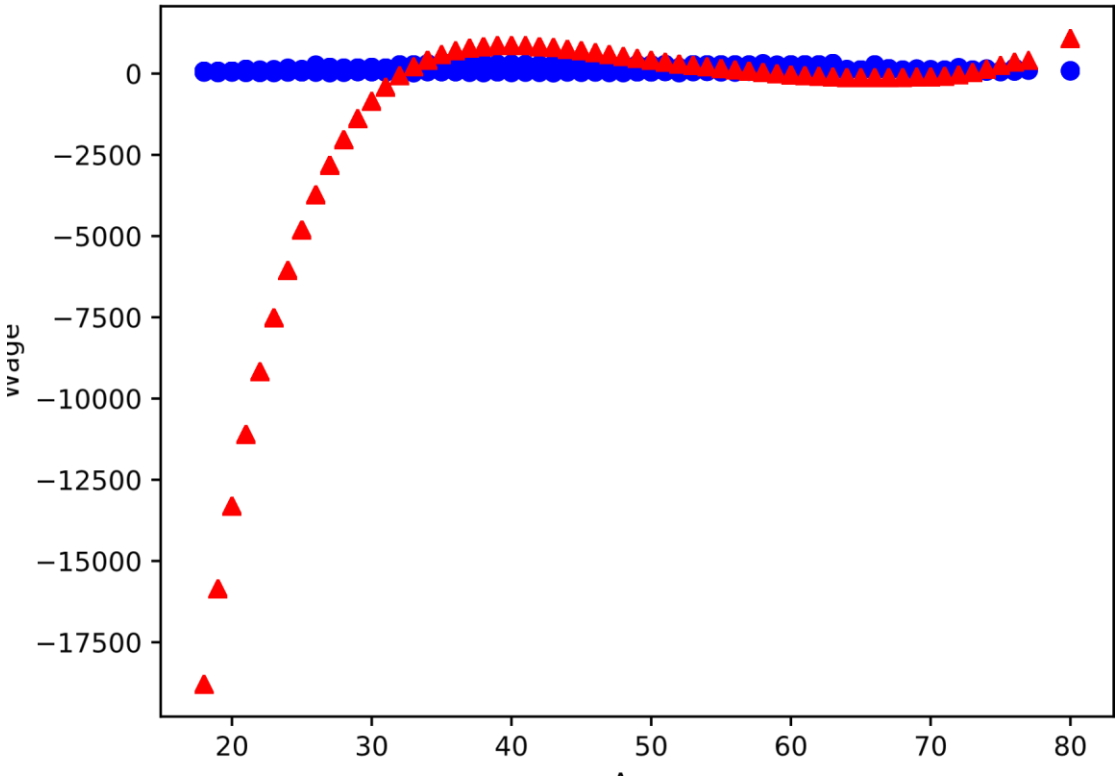
(2) Degree of polynomial = 5



(3) Degree of polynomial = 8

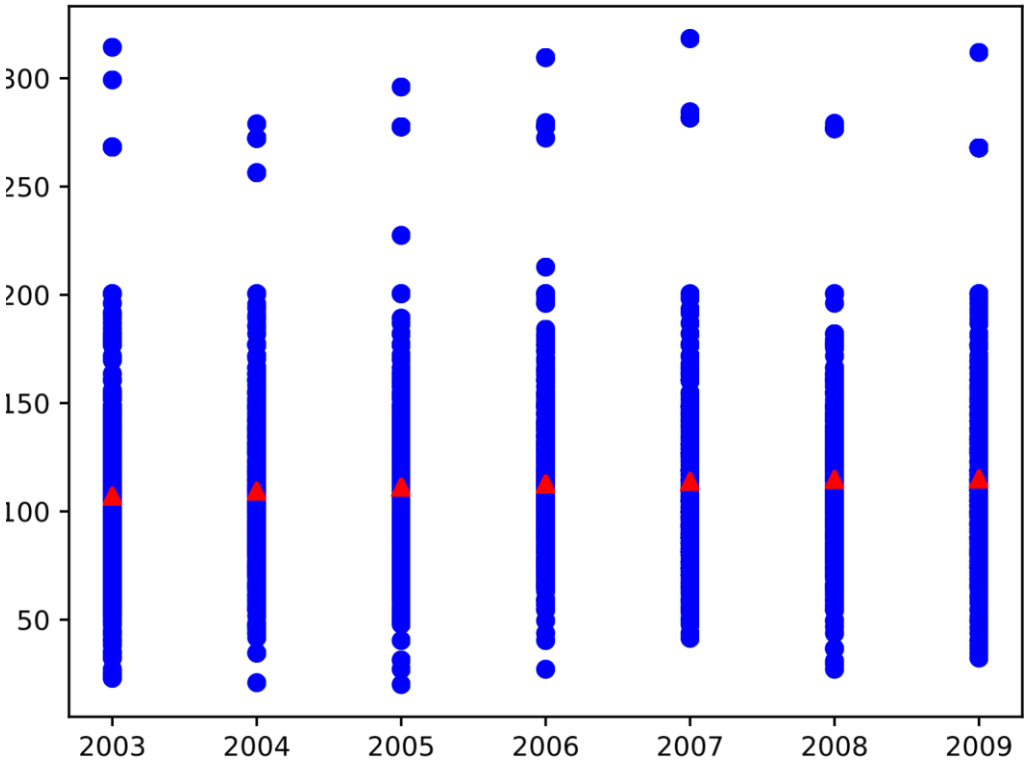


(4) Degree of polynomial = 25

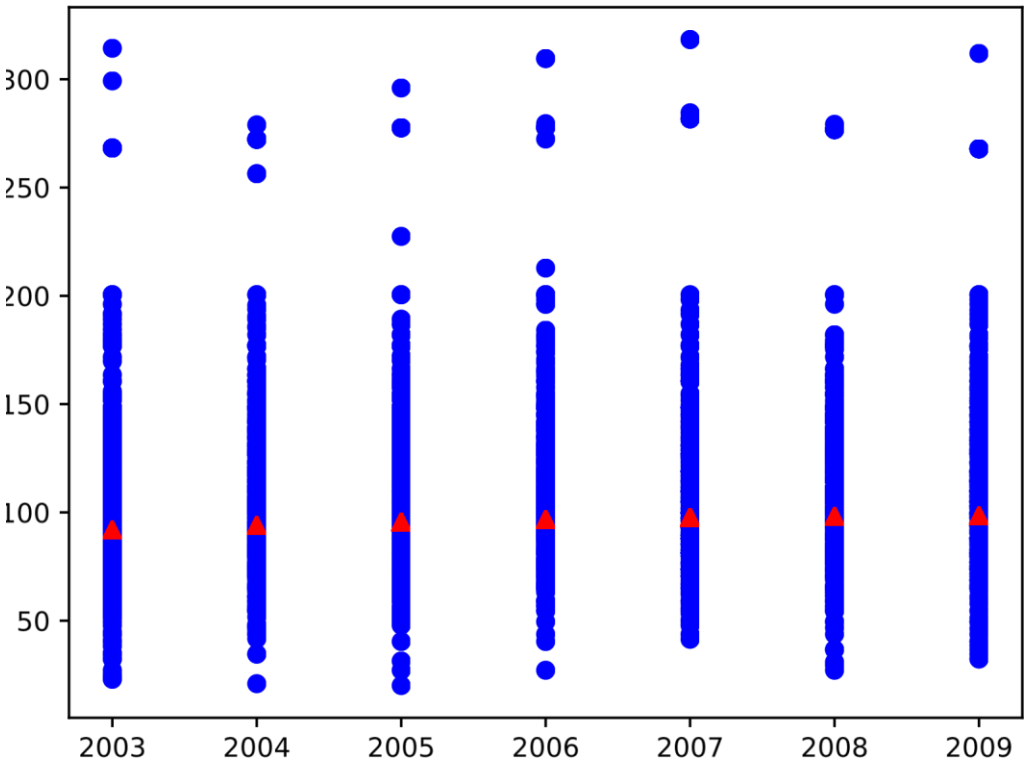


Wage vs Year plot(y = wage x = Year).

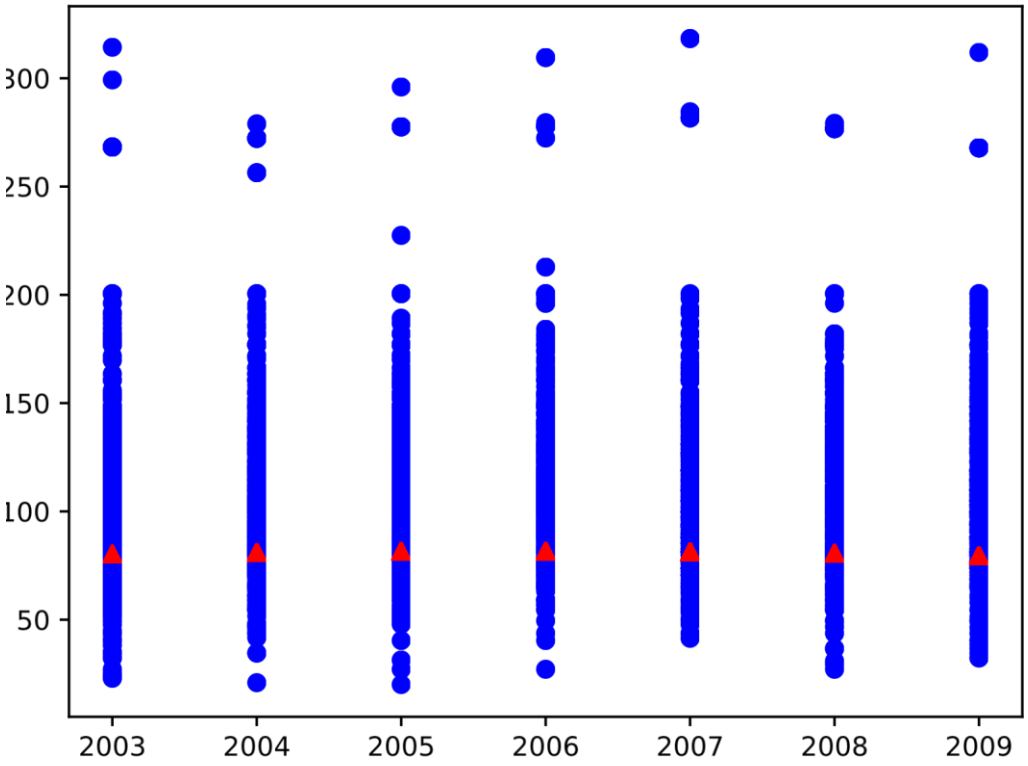
(1) Degree of polynomial = 2



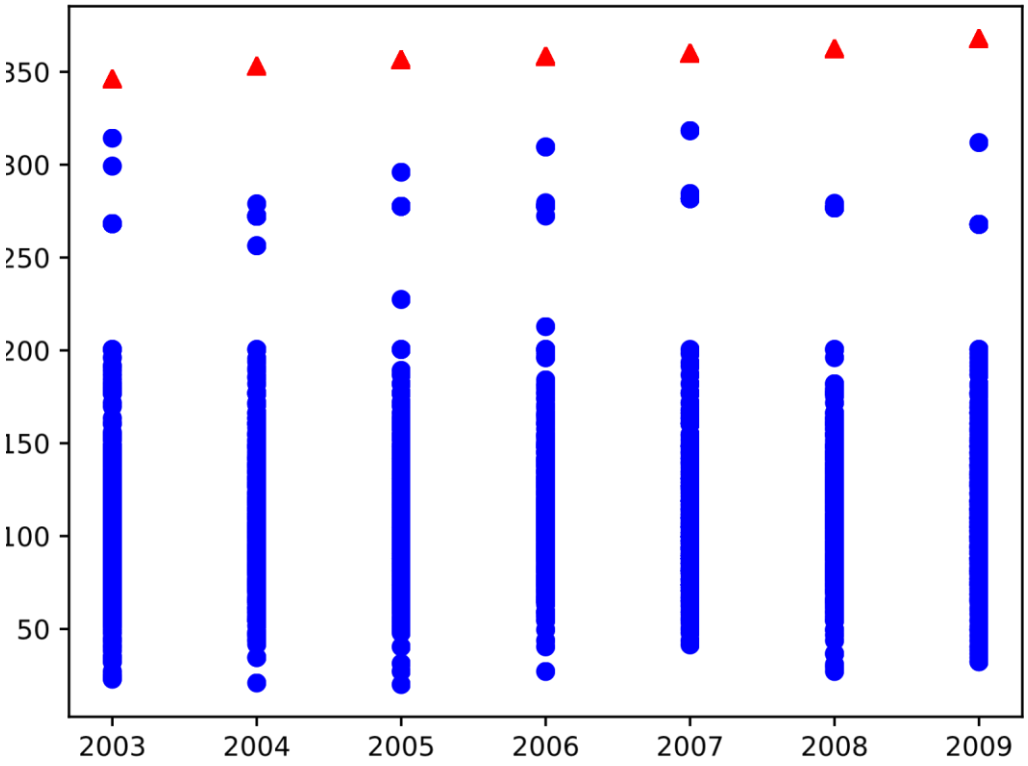
(2) Degree of polynomial = 5



(3) Degree of polynomial = 8

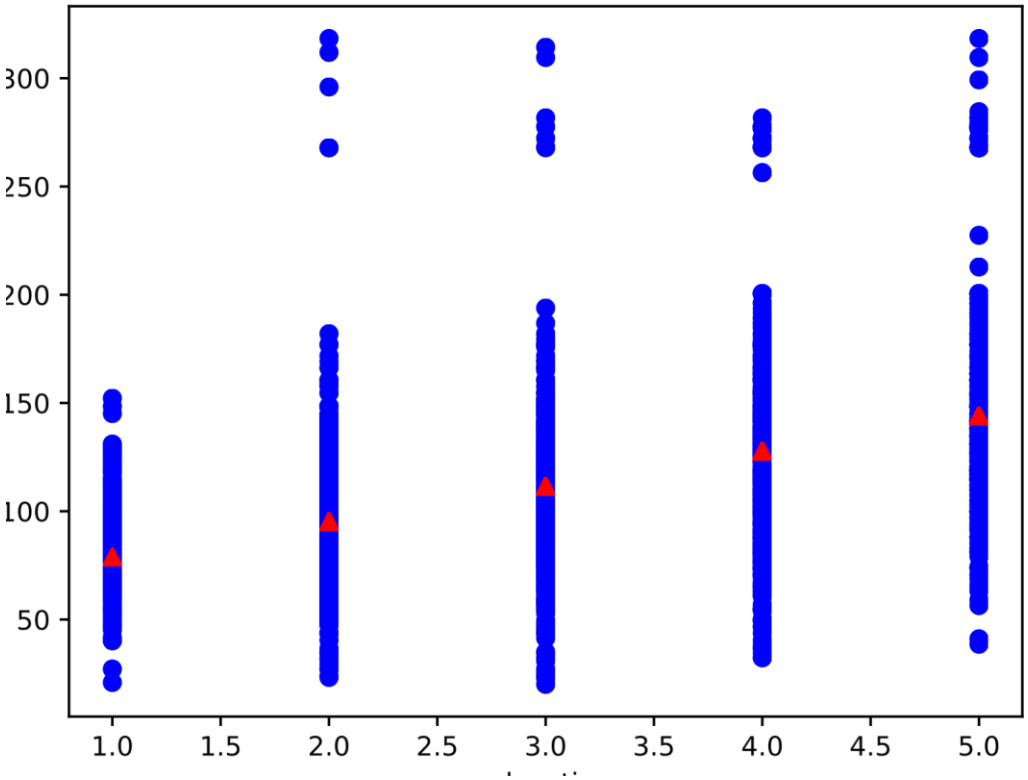


(4) Degree of polynomial = 10

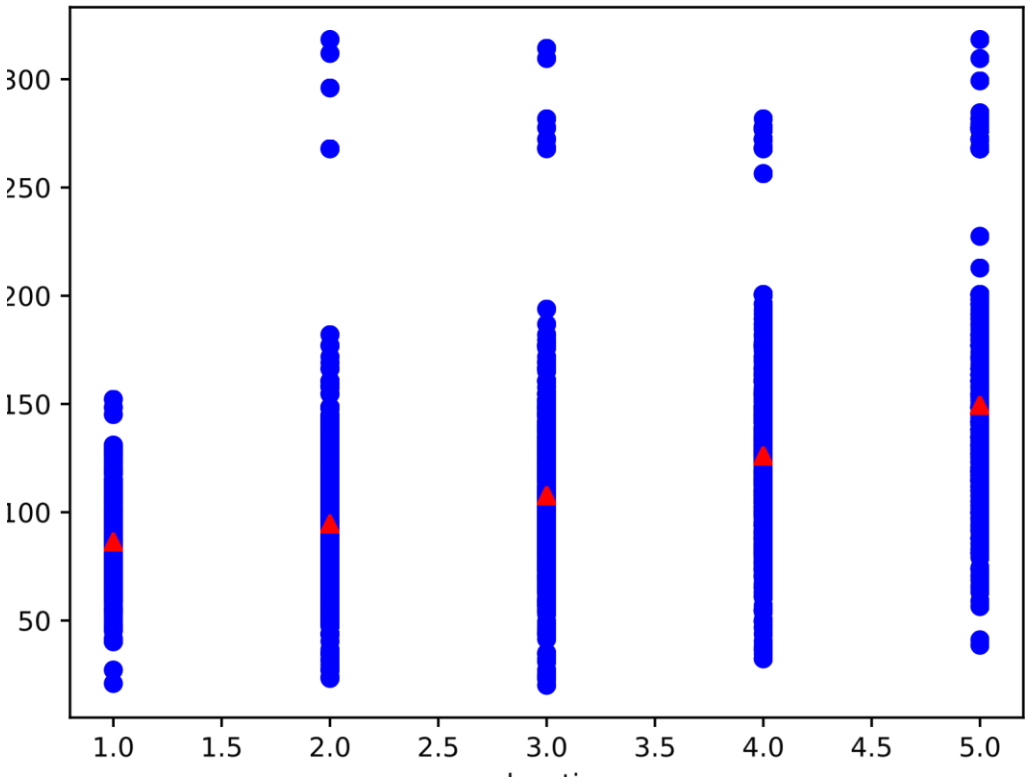


Wage vs education plot(y=wage x=education).

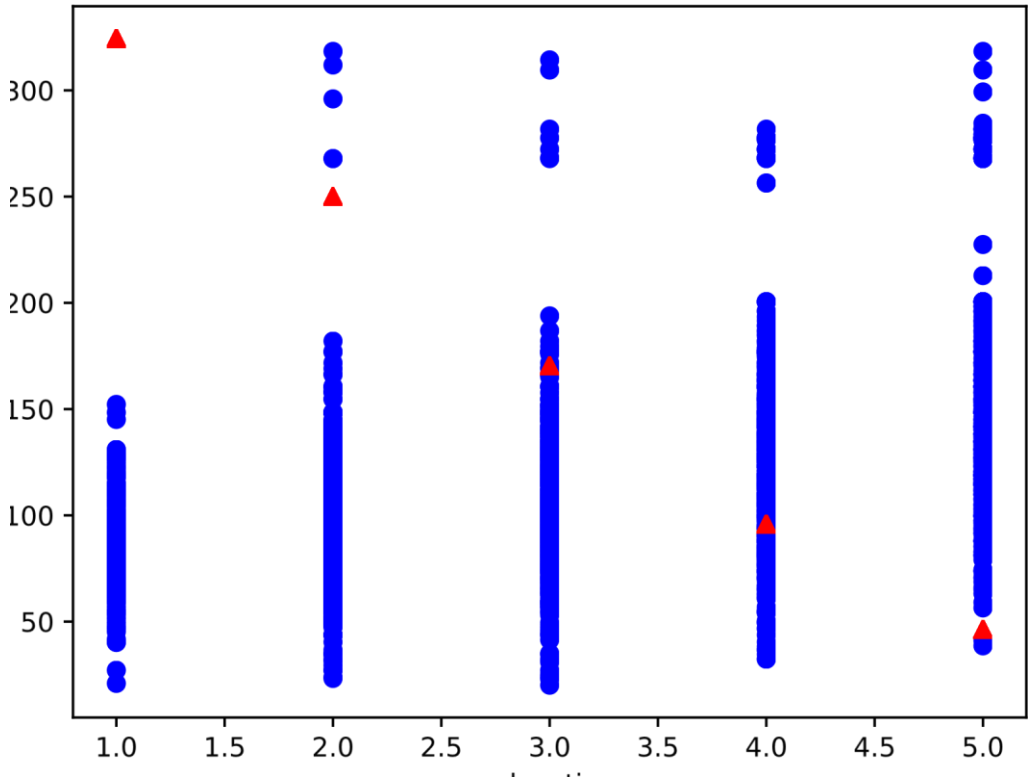
(1) Degree of polynomial = 1



(2) Degree of polynomial = 2



(3) Degree of polynomial = 5



Conclusion.

Wage vs Age:-

Increasing degree of polynomial, data is fitting well but after 7 or 8 degree data plot is seen to be overfit. As we can see from plot that at degree 25, ploynomial is completely overfitted. So it will be better to take degree 5 to fit the model as it can predict well on test data also.

Wage vs Year:-

On increasing degree of polynomial, fitted line is going downward which can be seen from the plot but at 10 degree it suddenly jumps to upper part which seems to be under fit.

So for this model, 2 degree polynomial can best fit the data as at 2 degree fitted model is passing to the denser data.

Wage vs Education:-

On increasing the degree of polynomial, it seems to be model is becoming overfit but at degree 5 plot suddenly changes and it becomes underfit.

So for this model, 1 degree polynomial will be better fit.

From all the three model I will take model 1 which is Wage as function of Age as data in it is continuous and also plot at degree 5 defines the data well by passing to the denser region.