REPORT

F

Problem 1

Problem: Learn the parameters μ_5, μ_6, $\sum_5$, $\sum_6$ and π by maximizing the likelihood,

$$\pi = Pr(r = C_5) \qquad Pr(x|C_5) = \mathcal{N}(x|\mu_5, \sum_5)$$
$$1 - \pi = Pr(r = C_6) \qquad Pr(x|C_6) = \mathcal{N}(x|\mu_6, \sum_6)$$

Use Bayes decision criterion to classify the test data. Estimate the misclassification rates of both classes and populate the 2x2 confusion matrix.

In problem 1 training set and test set is given from training set we have to make a model. In training set there are two classes label as 5 and 6 and 777 instances(rows) and 64 attributes(cols)
Our main aim is to find the Gaussian parameter mean and covariance from given training set for both of the classes and use these parameters to classify the test data. Estimate the misclassification rate and confusion matrix.

How to classify:-

By using Bayesian Decision Theory we will find classify the test data.

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:

$$P(Ci|x)=P(Ci)P(x|Ci)/P(x)$$

Where P(Ci|x) is Posterior , P(Ci) is Prior , P(x|Ci) is likelihood of i class.
P(x) we can calculate by using total probability. P(Ci) can be calculated from test data and label . Maximum p(Ci|x) means P(x|Ci) maximum so we have maximise likelihood.

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

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

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

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.

After finding mean and covariance from training set then find P(C5|x) and P(C6|x) for all x in test data and classify x by Bayesian decision :

Choose :- C5 if P(C5|x) > P(C6 I x)

C6 Else

Now we have label by using our model and actual label . Check for misclassification and estimate misclassification matrix.

Code Description :-

row_train = row of training set

Col_train = cols of training set

Mean5, mean6 are the mean of of Class 5 and Class 6 respectively from training set.

Similarly cov5 and cov6 are the covariance of class 5 and class 6 respectively from training set.

Calculation for mean and covariance:-

```
mean5=np.mean(X5,axis=0)
mean6=np.mean(X6,axis=0)
cov5=np.matmul(np.transpose(X5-mean5),(X5-mean5))/(no_c5-1)
cov6=np.matmul(np.transpose(X6-mean6),(X6-mean6))/(no_c6-1)
```

First I calculate mean and covariance from training set. And then find likelihood from test data to classify test data .

PxC5 is likelihood for class 5 and PxC6 is likelihood for class 6

PC5x is posterior for class 5 and PC6x is posterior for class 6.

After finding PC5x and PC6x made label for each x in test data

And then I checked label from test data and calculated label from model and found misclassification matrix

Calculation for posterior and misclassification matrix :-

```
P_c5=no_c5/row_train
P_c6=no_c6/row_train

cov5_inv=np.linalg.inv(cov5)
cov5_det=m.sqrt(abs(np.linalg.det(cov5)))

cov6_inv=np.linalg.inv(cov6)
cov6_det=m.sqrt(abs(np.linalg.det(cov6)))

pi_d=(m.sqrt(2*m.pi))**col_train
row_test=len(ts_data)

conf_mat=[[0,0],[0,0]]

for i in range(0,row_test):
    x=ts_data[i]-mean5
    g=np.matmul(x,cov5_inv)
    h=np.matmul(g,np.transpose(x))
    PxC5=np.exp(-h/2)/(pi_d*cov5_det)

    x=ts_data[i]-mean6
    g=np.matmul(x,cov6_inv)
    h=np.matmul(g,np.transpose(x))
    PxC6=np.exp(-h/2)/(pi_d*cov6_det)

    PC5x=PxC5*P_c5/(PxC5*P_c5+PxC6*P_c6)
    PC6x=PxC6*P_c6/(PxC5*P_c5+PxC6*P_c6)

    if(PC5x>PC6x and ts_label[i]==5):
        conf_mat[0][0]=conf_mat[0][0]+1
    elif(PC5x>PC6x and ts_label[i]==6):
        conf_mat[0][1]=conf_mat[0][1]+1
    elif(PC5x<PC6x and ts_label[i]==5):
        conf_mat[1][0]=conf_mat[1][0]+1
    else:
        conf_mat[1][1]=conf_mat[1][1]+1

print("")
```

Result:-

1 : cov5(covariance of class 5) is not equal to cov6(covariance of class6)

2: cov5 = cov6

3: cov6=cov5

```
deepak@deepak-Inspiron-3542:~/Desktop/6th sem/PR and ML/Assignment1$ python3 p1.py
2X2 confusion matrix for cov5!=cov6
[[106  27]
 [ 49 151]]
Percentage for cov5!=cov6
[[ 68.38709677  15.16853933]
 [ 31.61290323  84.83146067]]
deepak@deepak-Inspiron-3542:~/Desktop/6th sem/PR and ML/Assignment1$ python3 p1.py
2X2 confusion matrix for cov5!=cov6

[[106  27]
 [ 49 151]]

Percentage for cov5!=cov6
[[ 68.38709677  15.16853933]
 [ 31.61290323  84.83146067]]
deepak@deepak-Inspiron-3542:~/Desktop/6th sem/PR and ML/Assignment1$ python3 p1.py

2X2 confusion matrix for cov5!=cov6
[[106  27]
 [ 49 151]]

Percentage for case cov5!=cov6
[[ 68.38709677  15.16853933]
 [ 31.61290323  84.83146067]]
```

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:-

Problem: Learn a binary classifier for the given data taking class conditional densities as normal density. Estimate the misclassification rates of both classes, plot the discriminant function and iso-probability contours for the following cases:

Problem two is similar to problem one just we have to plot the discriminant function and iso-probability contours for the different cases:

Result for actual covariance from training data (cov0 and cov1)

Cov0 =

0.10758	0.1068
0.1068	7.0493

Cov1 =

1.0373	0.5788
0.5788	1.2807

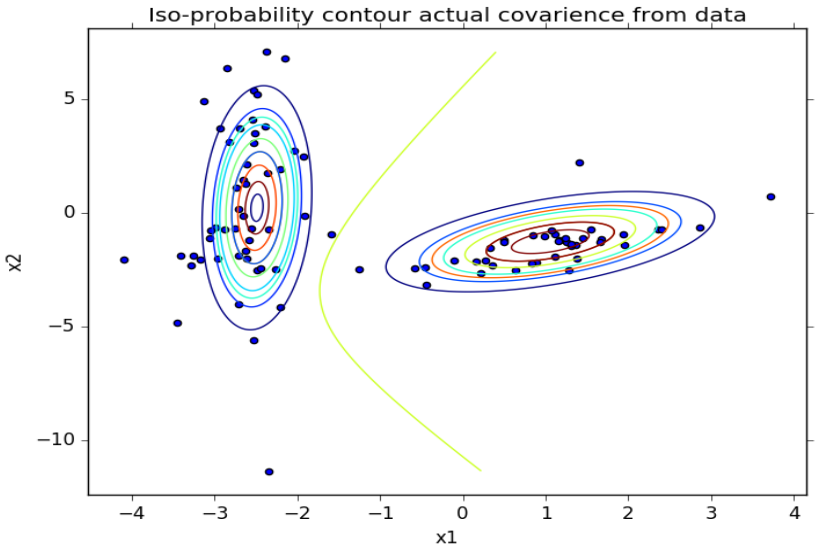
Confusion matrix =

50	0
0	40

Rate matrix =

100%	0%
0%	100%

Plot for Actual covariance :-



Code description for contour plot:-

```
def bi_normal(position, mean, covar):
    n = mean.shape[0]
    covar_det = np.linalg.det(covar)
    covar_inv = np.linalg.inv(covar)
    d = np.sqrt((2*np.pi)**n * covar_det)
    fac = np.einsum('...k,kl,...l->...', position-mean, covar_inv, position-mean)
    return np.exp(-fac / 2) / d
"""----------function end-----------------------------"""
x_range=np.linspace(min(ts_data[:,0]),max(ts_data[:,0]),1000)
y_range=np.linspace(min(ts_data[:,1]),max(ts_data[:,1]),1000)
X, Y = np.meshgrid(x_range,y_range)

d = np.empty(X.shape + (2,))

d[:, :, 0] = X
d[:, :, 1] = Y
Z0 = bi_normal(d,mean0,cov0)
plt.contour(X,Y,Z0)
Z1 = bi_normal(d,mean1,cov1)
plt.contour(X,Y,Z1)


Z = (Z1 - Z0)
plt.contour(X,Y,Z)

plt.title('Iso-probability contour (case4 a1=3,b1=4,c1=2,d1=7/ a2=14,b2=10,c2=12,d2=16)')
plt.scatter(ts_data[:,0],ts_data[:,1])
plt.ylabel('x2')
plt.xlabel('x1')
plt.show()
```

Contour plot for following cases :-

(a) Equal diagonal $\sum_s$ of equal variances along both dimensions, $\sum_0 = \sum_1 = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}$

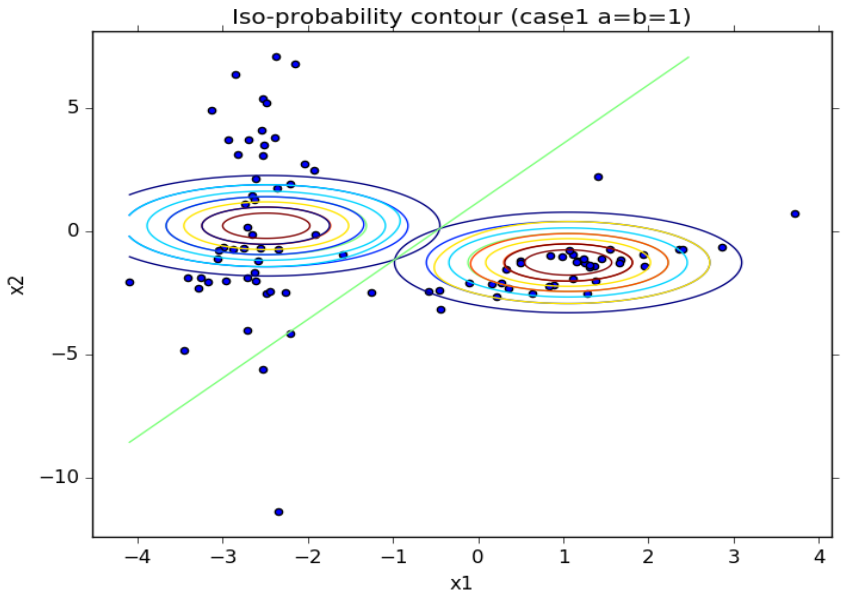
(b) Equal diagonal $\sum_s$ with unequal variances along different dimensions, $\sum_0 = \sum_1 = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$

(c) Arbitrary $\sum_s$ but shared by both classes, $\sum_0 = \sum_1 = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

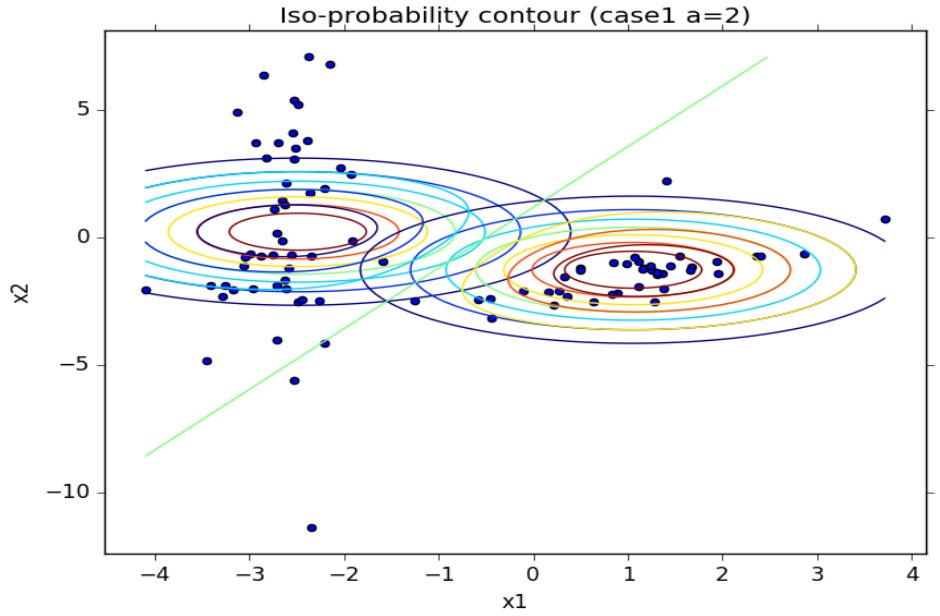
(d) Different arbitrary $\sum_s$ for the two classes.

Case 1 :- $\sum_0 = \sum_1 = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}$

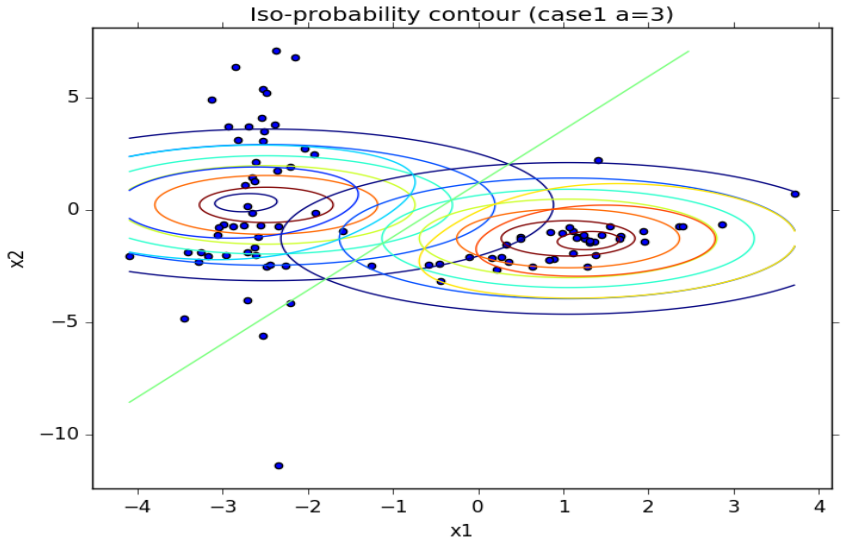
a=1



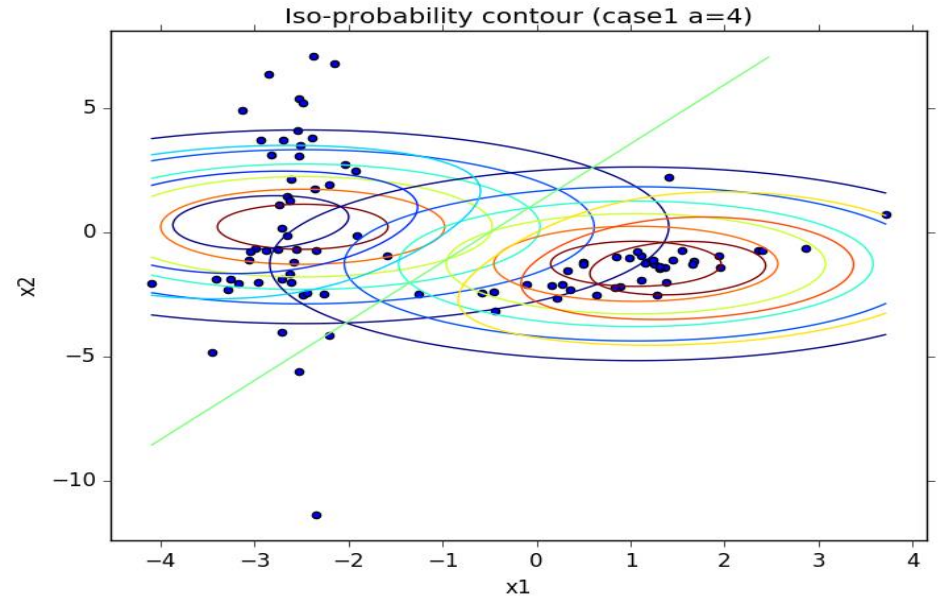
a=2



a=3



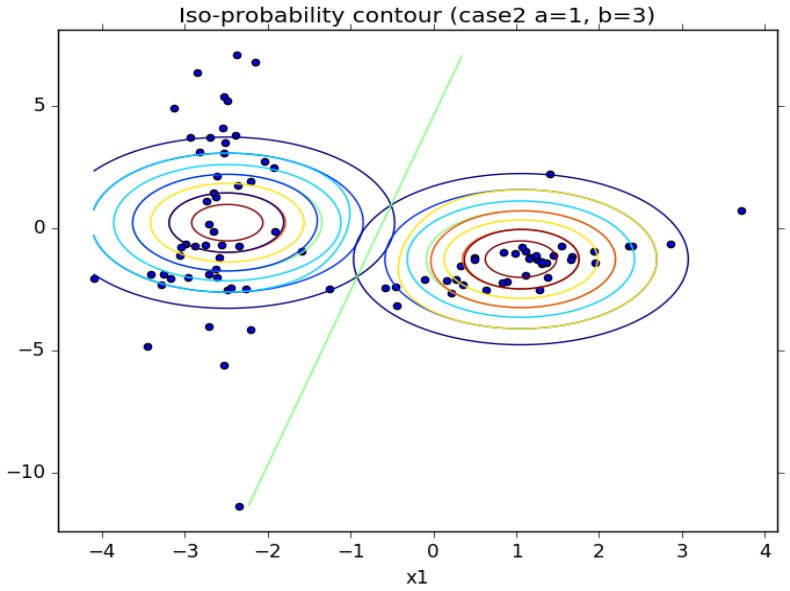
a=4



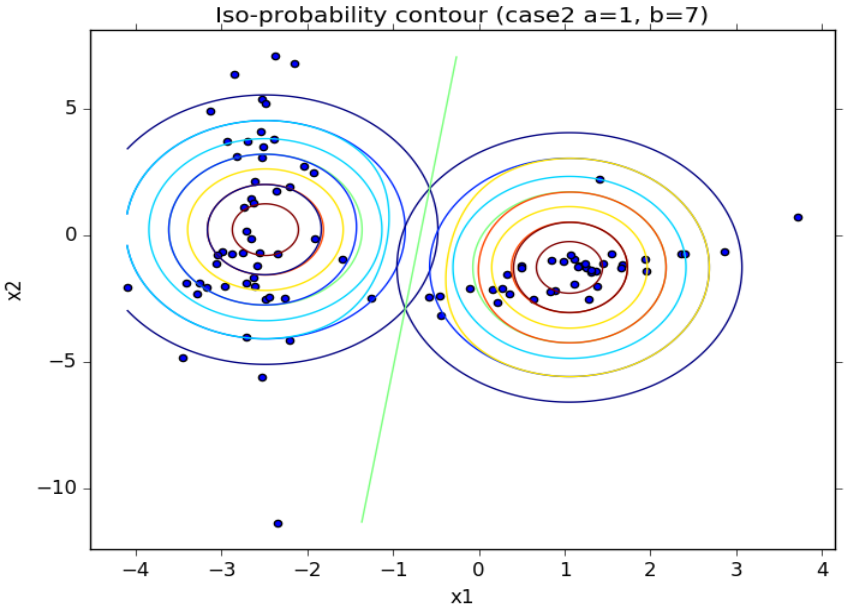
Case 2:-

$$\sum_0 = \sum_1 = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$$

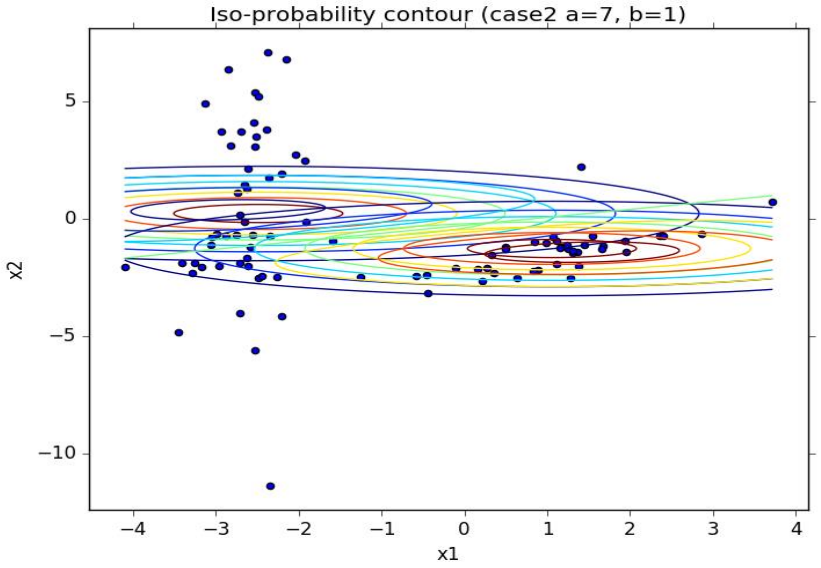
a=1, b=3



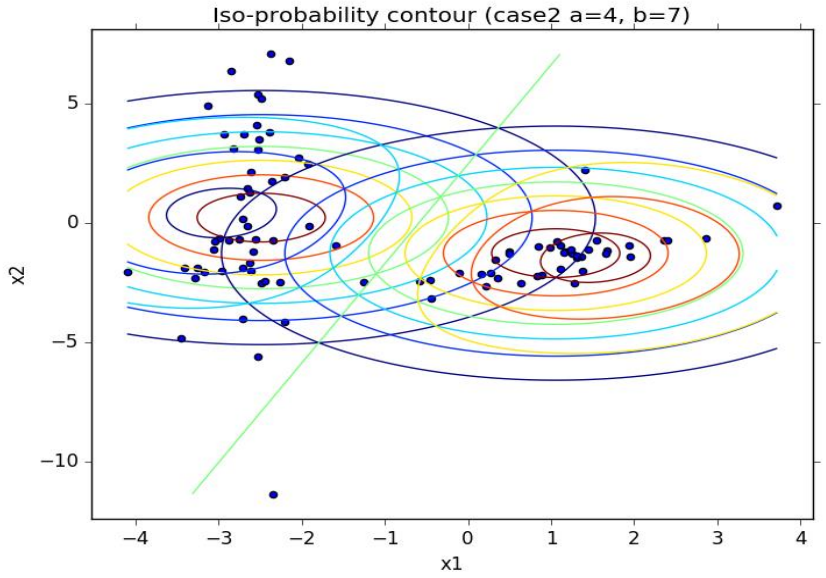
a=1,b=7



a=7,b=1

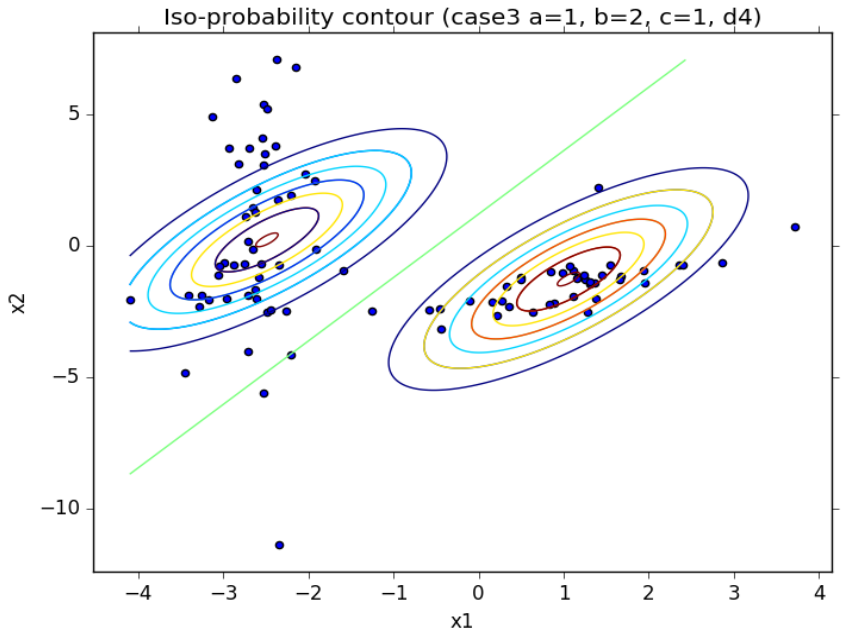


a=4,b=7

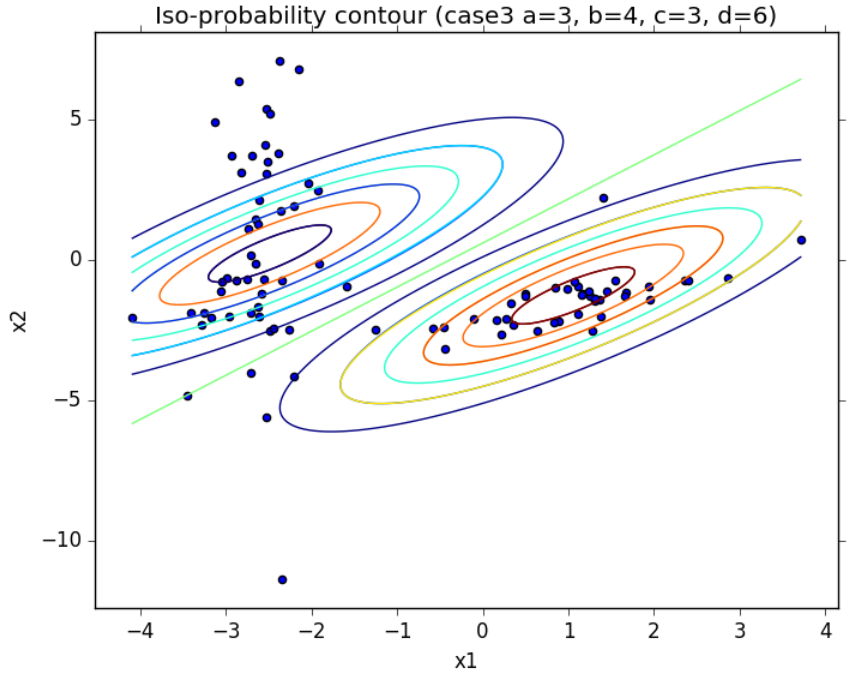


Case 3:-

$$\sum_0 = \sum_1 = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

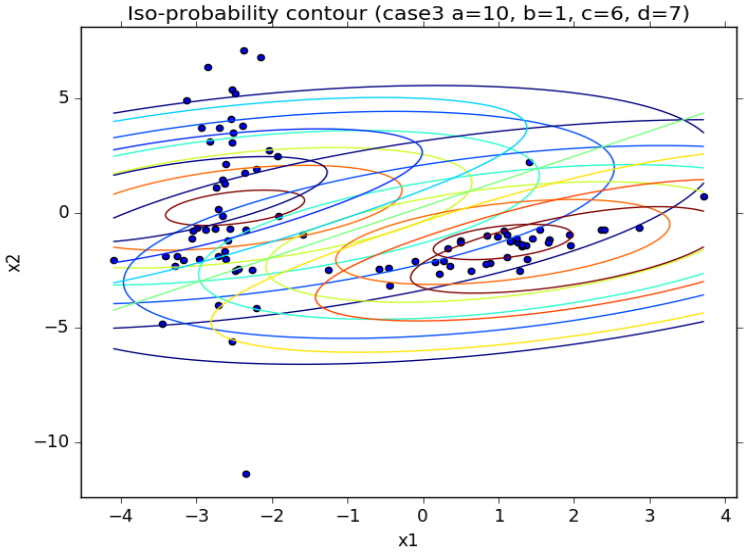


a=1,b=2,c=1,d=4

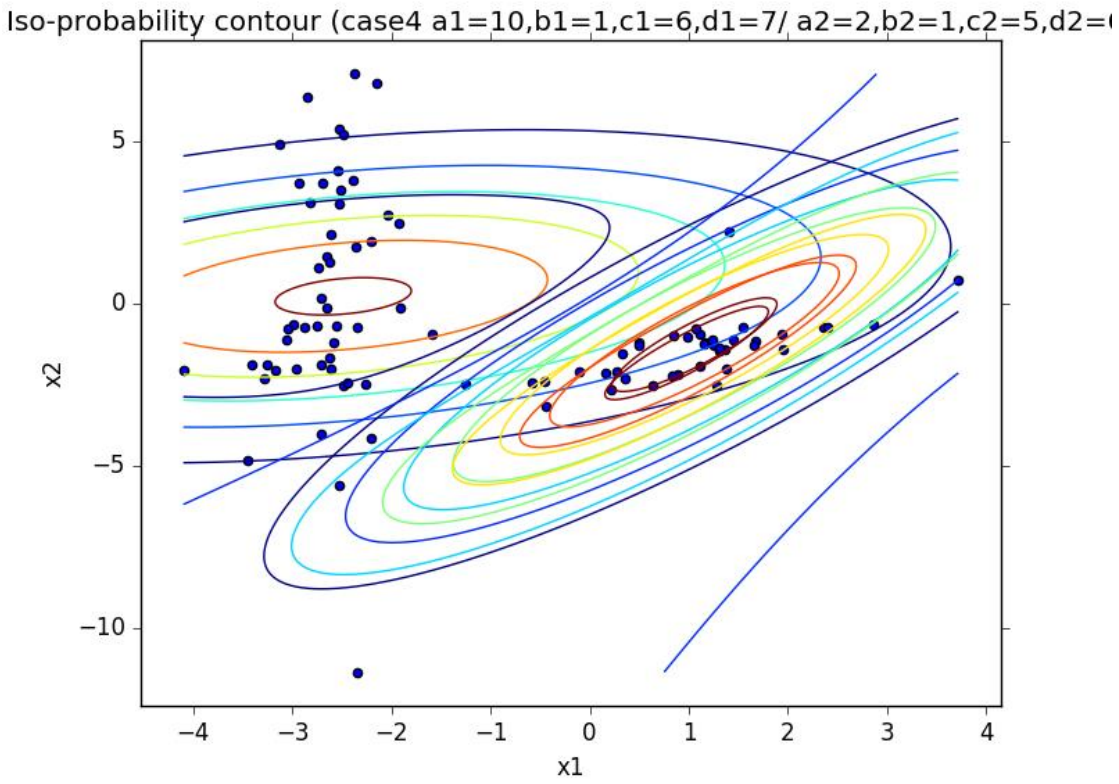


a=3,b=4,c=3d=6

a=10,b=1,c=6,d=7



Case 4:- Different arbitrary $\sum_s$ for the two classes.

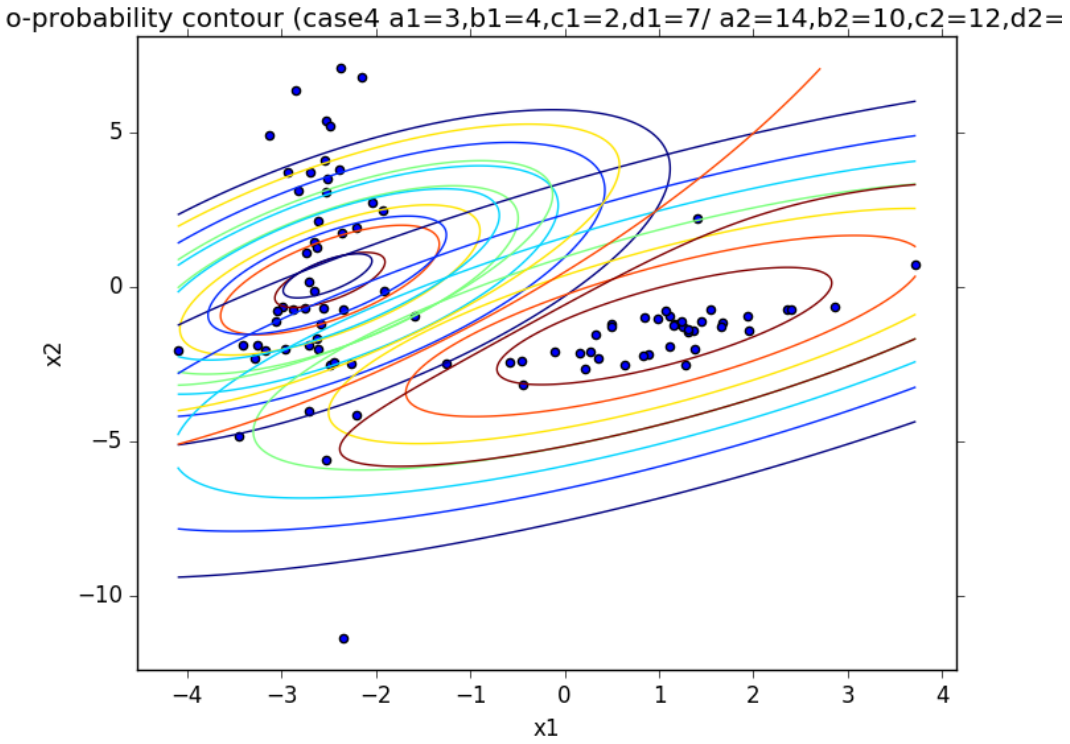


Cov0 =

10	1
6	7

Cov1 =

2	1
5	6



Cov0 =

3	4
2	7

Cov1 =

14	10
12	16

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 of overlapping of both are decreasing. For all variance for case 1 confusion matrix remain same.

Case2:- By keeping the value of ‘b’ constant and increasing value of ‘a’ we see from the plot that elliptical overlapping is increasing and decision boundary is shifting towards X1.

By keeping the value of ‘a’ constant and increasing value of ‘b’ we see from the plot that elliptical overlapping is increasing and decision boundary is shifting towards X2.

Case3:- I observe that the 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 the plot arbitrary values and also for the different covariance then I observe that these contours are combinations of above 3 cases.

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

Problem 3:-

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.

Summery :-

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.

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

N is number of data set.

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

Code Description:-

year is variable to store years.

age is to store ages of peoples

education is to store education status of people.

```
year=np.array(tr_data[:,0])
age=np.array(tr_data[:,1])
education=np.array(tr_data[:,4])
wage=np.array(tr_data[:,10])
```

A and y is given below :-

```
A=np.zeros((k,k))
y=np.zeros(k)

for i in range(0,k):
    y[i]=np.sum(wage*age**i)
    for j in range(0,k):
        A[i][j]=np.sum(age**(i+j))
W_age=np.matmul(np.linalg.inv(A),np.transpose(y))
```

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

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

W_age is A inverse multiply by Y.

W I am calculating parameter for model wage as function of age.

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

After that I am plotting Wage vs age plot.

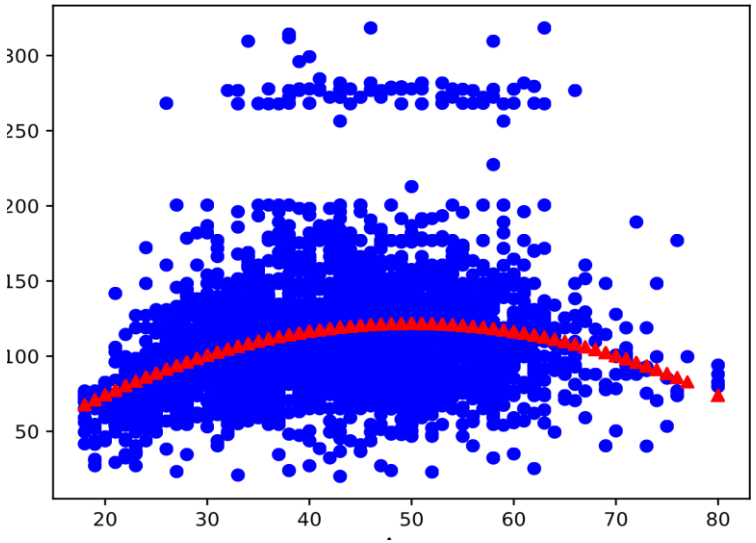
```
r_wage=[]
for i in range(0,n):
    sm=0
    for j in range(0,k):
        sm=sm+W_age[j]*age[i]**j
    r_wage.append(sm)
```

Similarly I did for year vs wage and education vs wage.

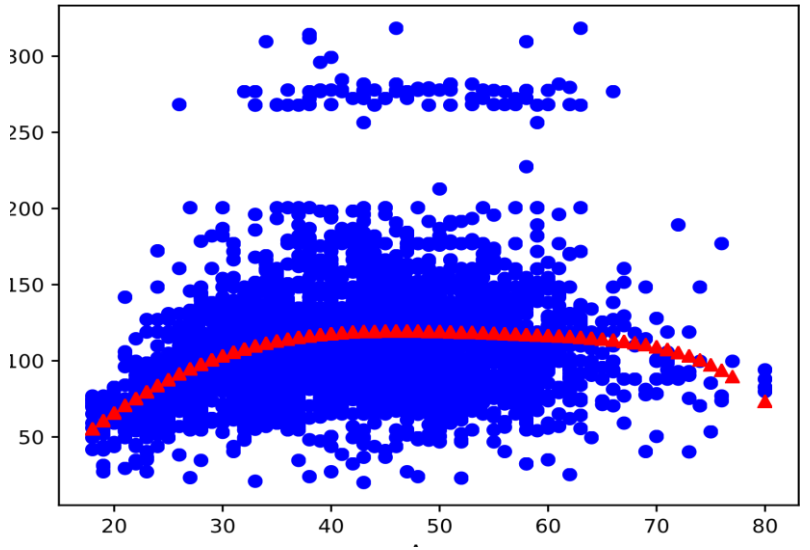
Result:-

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

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

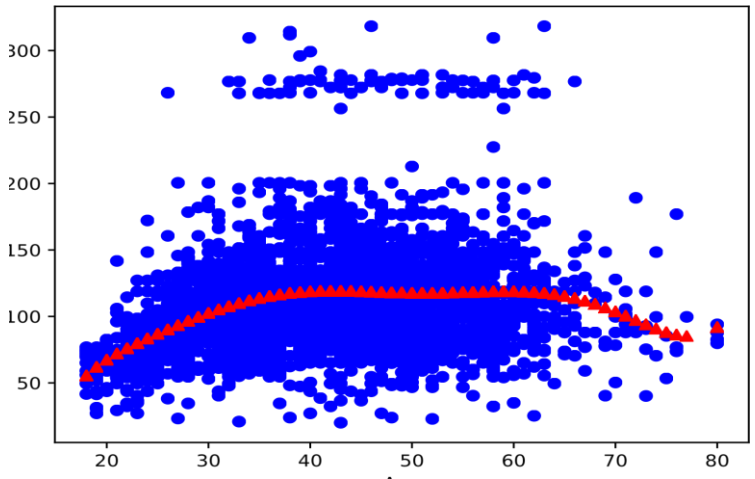


Degree of polynomial = 2

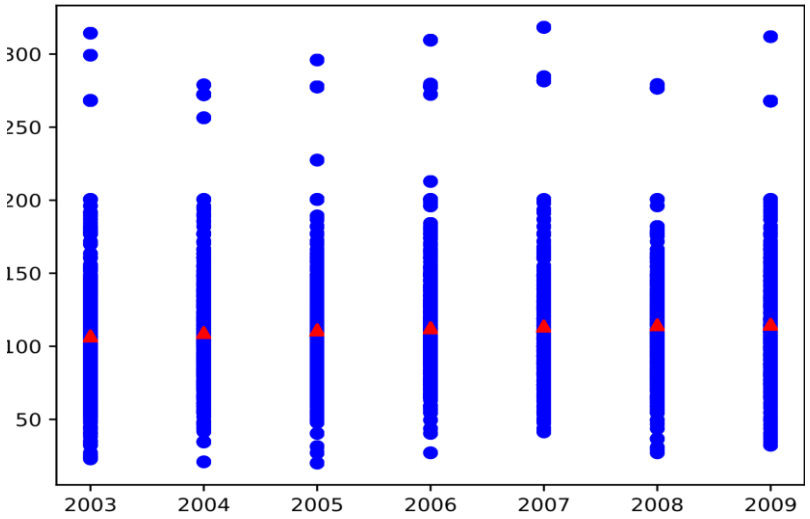


Degree of polynomial = 5

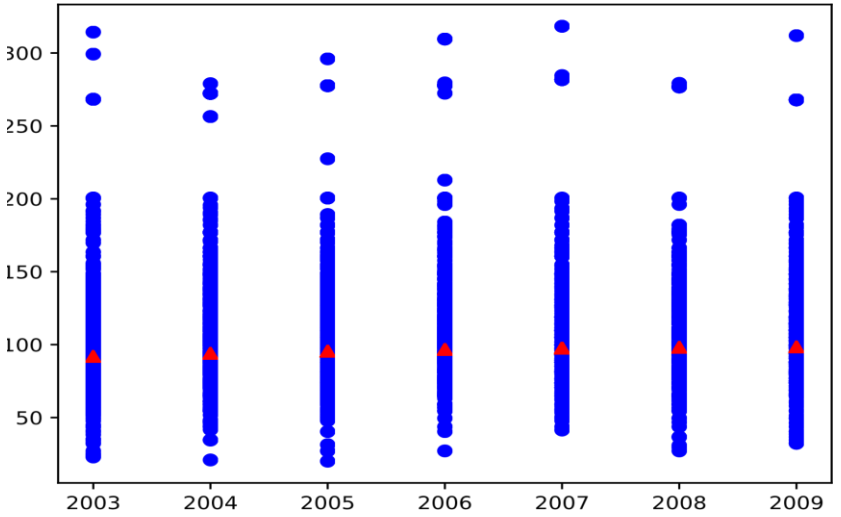
Degree of polynomial = 8



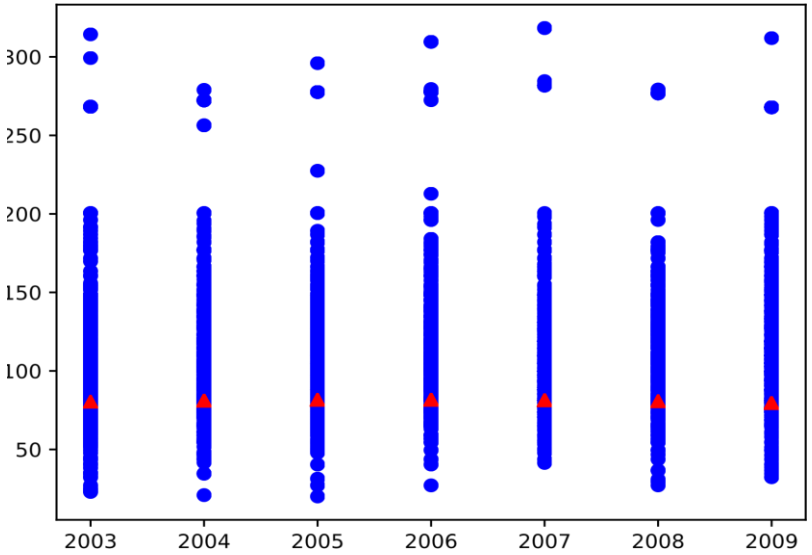
Wage vs Year plot(y = wage x = Year).



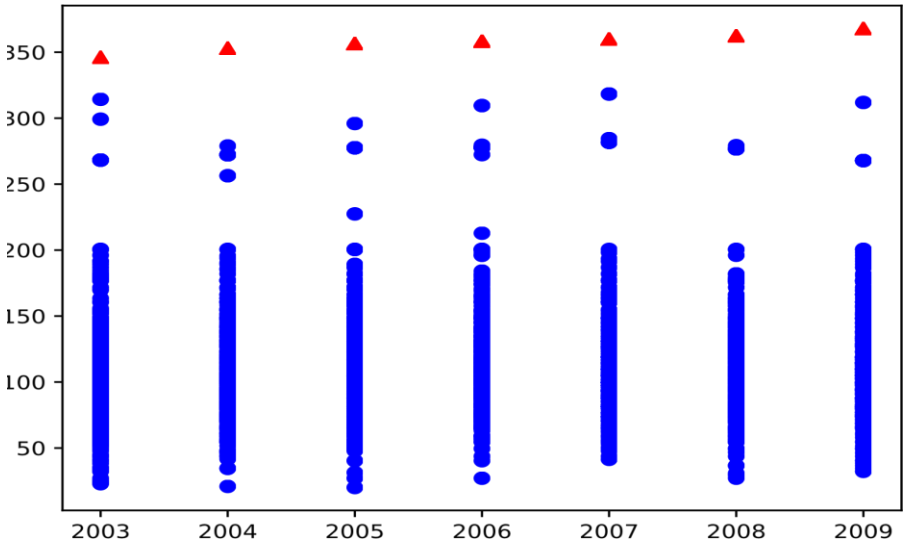
Degree of polynomial = 2



Degree of polynomial = 5

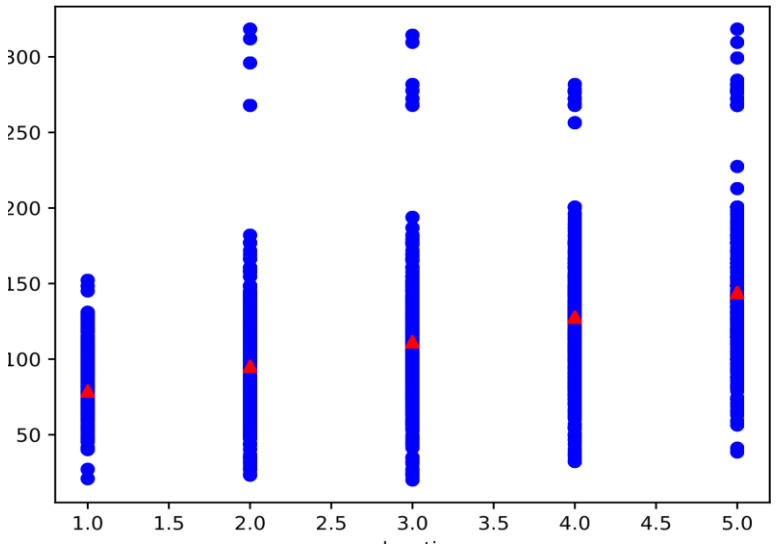


Degree of polynomial = 8

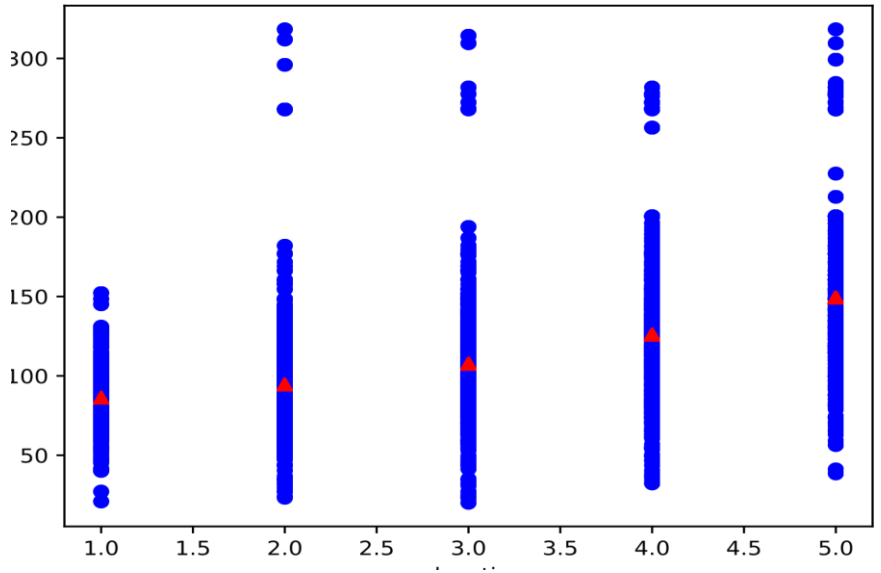


Degree of polynomial = 10

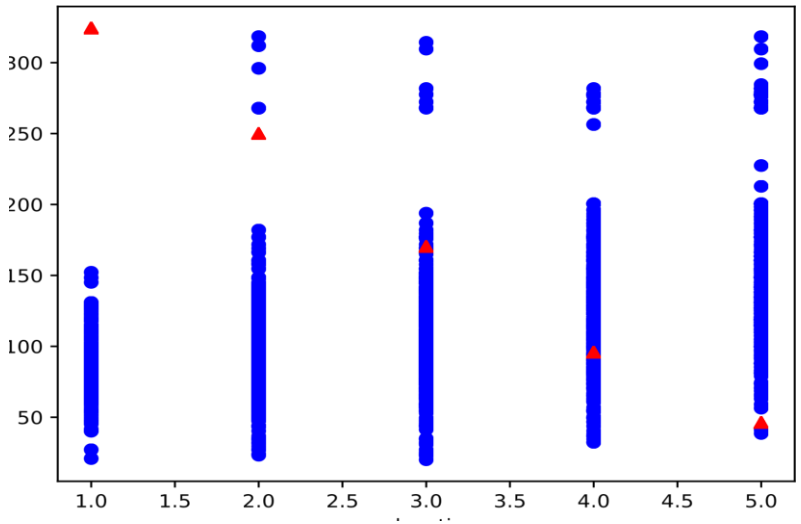
Wage vs education plot(y=wage x=education).



Degree of polynomial = 1



Degree of polynomial = 2



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.