**Name: Mohammad Zain Abbas**

**Reg #: 6865**

**DE-36 (CE), Syndicate: A**

**LAB 13 JOURNEL**

**Equipment Used:** Notebook Computer, Matlab , Visual Studio(C#)

**Lab Tasks:**

1) Use the given cancer dataset within MATLAB and classify it using Naïve Bayes classification

model:

a) First create a MATLAB script and load ‘cancer’ mat file.

b) Identify features and classes from the loaded dataset.

c) Perform 2-fold cross validation on the dataset by splitting it into testing and

training parts.

d) Implement a Bayesian classifier using the above algorithm and use training dataset

to classify each of the sample within testing dataset.

e) Compute the accuracy from the predicted test samples.

**SOLUTION CODE:**

warning off

clear all;

load ('cancer.mat')

[rows,cols]=size(dataset);

Training\_Dataset=dataset(1:rows/2,1:9);

Training\_Labels=dataset(1:rows/2,10);

Testing\_Dataset=dataset(rows/2:end-1,1:9);

Testing\_Labels=dataset(rows/2:end-1,10);

Testing\_Labels=Testing\_Labels';

counter0=1;

counter1=1;

[rows\_training,cols\_training]=size(Training\_Dataset);

for i=1:rows\_training

if(Training\_Labels(i)==0)

Class0(counter0,:)=Training\_Dataset(i,:);

counter0=counter0+1;

else

Class1(counter1,:)=Training\_Dataset(i,:);

counter1=counter1+1;

end

end

Prior\_Probability0=counter0/length(Training\_Labels);

Prior\_Probability1=counter1/length(Training\_Labels);

for i=1:cols\_training

mean\_Features\_Class0(i)=mean(Class0(:,i));

mean\_Features\_Class1(i)=mean(Class1(:,i));

var\_Feature\_Class0(i)=var(Class0(:,i));

var\_Feature\_Class1(i)=var(Class1(:,i));

end

for i=1:cols\_training

for j=1:length(Testing\_Dataset)

Likelihood\_ProbabilityFeatures\_Class0(i,j)=(1/sqrt(2\*pi\*var\_Feature\_Class0(i)))\*(exp(-((Testing\_Dataset(j,i)-mean\_Features\_Class0(i))^2)/(2\*var\_Feature\_Class0(i))));

Likelihood\_ProbabilityFeatures\_Class1(i,j)=(1/sqrt(2\*pi\*var\_Feature\_Class1(i)))\*(exp(-((Testing\_Dataset(j,i)-mean\_Features\_Class1(i))^2)/(2\*var\_Feature\_Class1(i))));

end

end

Posterior\_Probability0=Likelihood\_ProbabilityFeatures\_Class0(1,:).\*Likelihood\_ProbabilityFeatures\_Class0(2,:) .\* Likelihood\_ProbabilityFeatures\_Class0(3,:) .\* Likelihood\_ProbabilityFeatures\_Class0(4,:) .\* Likelihood\_ProbabilityFeatures\_Class0(5,:) .\* Likelihood\_ProbabilityFeatures\_Class0(6,:) .\* Likelihood\_ProbabilityFeatures\_Class0(7,:).\*Likelihood\_ProbabilityFeatures\_Class0(8,:).\* Likelihood\_ProbabilityFeatures\_Class0(9,:).\* Prior\_Probability0;

Posterior\_Probability1=Likelihood\_ProbabilityFeatures\_Class1(1,:).\*Likelihood\_ProbabilityFeatures\_Class1(2,:) .\* Likelihood\_ProbabilityFeatures\_Class1(3,:) .\* Likelihood\_ProbabilityFeatures\_Class1(4,:) .\* Likelihood\_ProbabilityFeatures\_Class1(5,:) .\* Likelihood\_ProbabilityFeatures\_Class1(6,:) .\* Likelihood\_ProbabilityFeatures\_Class1(7,:).\*Likelihood\_ProbabilityFeatures\_Class1(8,:).\* Likelihood\_ProbabilityFeatures\_Class1(9,:).\* Prior\_Probability1;

Evidance=Posterior\_Probability0+Posterior\_Probability1;

Normalized\_Posterior\_Probability0=Posterior\_Probability0./Evidance;

Normalized\_Posterior\_Probability1=Posterior\_Probability1./Evidance;

Predicted\_Labels=zeros(1,length(Normalized\_Posterior\_Probability1));

for k=1:length(Normalized\_Posterior\_Probability0)

if(Normalized\_Posterior\_Probability0(k)>=Normalized\_Posterior\_Probability1(k))

Predicted\_Labels(k)=0;

else

Predicted\_Labels(k)=1;

end

end

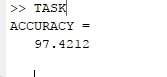
Accuracy=sum(Predicted\_Labels==Testing\_Labels);

Accuracy=(Accuracy/length(Testing\_Labels)\*100);

disp('ACCURACY = ');

disp(Accuracy);

**OUTPUT:**



2.Develop a C# console application to implement NaiveBayes classification model for the

following dataset and classify the given test vector:

**SOLUTION CODE:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LAB13

{

public class FeatureVector

{

public int Age;

public int Loan;

public char Defaulter;

public FeatureVector(int age, int loan,char defaulter)

{

this.Age = age;

this.Loan = loan;

this.Defaulter = defaulter;

}

}

class Program

{

public static void Main(string[] args)

{

float prior0 = 6 / 11f;

float prior1 = 5 / 11f;

List<FeatureVector> Train\_Dataset = new List<FeatureVector>();

List<FeatureVector> Test\_Dataset = new List<FeatureVector>();

Train\_Dataset.Add(new FeatureVector(25, 40000, '0'));

Train\_Dataset.Add(new FeatureVector(35, 60000, '0'));

Train\_Dataset.Add(new FeatureVector(45, 80000, '0'));

Train\_Dataset.Add(new FeatureVector(20, 20000, '0'));

Train\_Dataset.Add(new FeatureVector(35, 120000, '0'));

Train\_Dataset.Add(new FeatureVector(52, 18000, '0'));

Train\_Dataset.Add(new FeatureVector(23, 95000, '1'));

Train\_Dataset.Add(new FeatureVector(40, 62000, '1'));

Train\_Dataset.Add(new FeatureVector(60, 100000, '1'));

Train\_Dataset.Add(new FeatureVector(48, 220000, '1'));

Train\_Dataset.Add(new FeatureVector(33, 150000, '1'));

Test\_Dataset.Add(new FeatureVector(48, 120000, '?'));

float Mean\_AgeFeature0 = mean\_AgeFeature('0', Train\_Dataset);

float Mean\_AgeFeature1 = mean\_AgeFeature('1', Train\_Dataset);

float Mean\_LoanFeature0 = mean\_LoanFeature('0', Train\_Dataset);

float Mean\_LoanFeature1 = mean\_LoanFeature('1', Train\_Dataset);

float Variance\_AgeFeature0 = variance\_AgeFeature('0', Mean\_AgeFeature0, Train\_Dataset);

float Variance\_LoanFeature0 = variance\_LoanFeature('0', Mean\_LoanFeature0, Train\_Dataset);

float Variance\_AgeFeature1 = variance\_AgeFeature('1', Mean\_AgeFeature1, Train\_Dataset);

float Variance\_LoanFeature1 = variance\_LoanFeature('1', Mean\_LoanFeature1, Train\_Dataset);

var std00 = (float)Math.Sqrt(Variance\_AgeFeature0);

var std01 = (float)Math.Sqrt(Variance\_LoanFeature0);

var std10 = (float)Math.Sqrt(Variance\_AgeFeature1);

var std11 = (float)Math.Sqrt(Variance\_LoanFeature1);

float Likelihood\_Age0 = Likelihood\_Probability(Test\_Dataset[0].Age, '0', Mean\_AgeFeature0, std00);

float Likelihood\_Loan0 = Likelihood\_Probability(Test\_Dataset[0].Loan, '0', Mean\_LoanFeature0, std01);

float Likelihood\_Age1 = Likelihood\_Probability(Test\_Dataset[0].Age, '1', Mean\_AgeFeature1, std10);

float Likelihood\_Loan1 = Likelihood\_Probability(Test\_Dataset[0].Loan, '1', Mean\_LoanFeature1, std11);

float Posterior0 = Posterior\_Probability(Likelihood\_Age0,Likelihood\_Loan0, '0', prior0);

float Posterior1 = Posterior\_Probability(Likelihood\_Age1, Likelihood\_Loan1, '1', prior1);

float evidence = Posterior0 + Posterior1;

float normalized\_Probability0 = Posterior0 / evidence;

float normalized\_Probability1 = Posterior1 / evidence;

Console.WriteLine("Normalized Probability of 0" + ": " + normalized\_Probability0);

Console.WriteLine("Normalized Probability of 1" + ": " + normalized\_Probability1);

if (normalized\_Probability0 > normalized\_Probability1)

{

Console.WriteLine("Sample is assigned to Class 0");

}

else

{

Console.WriteLine("Sample is assigned to Class 1");

}

}

public static float mean\_AgeFeature(char classVal, List<FeatureVector> TrainData)

{

float mean = 0;

float sum = 0;

int count = 0;

for (int c = 0; c < TrainData.Count; c++)

{

if (TrainData[c].Defaulter == classVal)

{

sum = sum + TrainData[c].Age;

count++;

}

}

mean = sum / count;

Console.WriteLine("Mean of Age Feature " + classVal + ": " + mean);

return mean;

}

public static float mean\_LoanFeature(char classVal, List<FeatureVector> TrainData)

{

float mean = 0;

float sum = 0;

int count = 0;

for (int c = 0; c < TrainData.Count; c++)

{

if (TrainData[c].Defaulter == classVal)

{

sum = sum + TrainData[c].Loan;

count++;

}

}

mean = sum / count;

Console.WriteLine("Mean of Loan Feature" + classVal + ": " + mean);

return mean;

}

public static float variance\_AgeFeature(char classVal, float meanVal, List<FeatureVector> TrainData)

{

float variance = 0;

float count = 0;

float sum = 0;

for (int c = 0; c < TrainData.Count; c++)

{

if (TrainData[c].Defaulter == classVal)

{

sum = sum + (float)Math.Pow((TrainData[c].Age - meanVal), 2);

count++;

}

}

variance = sum / count;

Console.WriteLine("Variance of Age Feature " + classVal + ": " + variance);

return variance;

}

public static float variance\_LoanFeature(char classVal, float meanVal, List<FeatureVector> TrainData)

{

float variance = 0;

float count = 0;

float sum = 0;

for (int c = 0; c < TrainData.Count; c++)

{

if (TrainData[c].Defaulter == classVal)

{

sum = sum + (float)Math.Pow((TrainData[c].Loan - meanVal), 2);

count++;

}

}

variance = sum / count;

Console.WriteLine("Variance of Loan Feature " + classVal + ": " + variance);

return variance;

}

public static float Likelihood\_Probability(int TestSample, char classVal, float meanVal, float Variace\_Val)

{

float Probability = 0;

Probability = (float)(1 / Math.Sqrt(2 \* Math.PI) \* Variace\_Val) \* (float)Math.Exp((-1 \* (float)Math.Pow((TestSample - meanVal), 2)) / (2 \* Math.Pow(Variace\_Val, 2)));

Console.WriteLine("Likelihood Probability " + classVal + ": " + Probability);

return Probability;

}

public static float Posterior\_Probability(float Likelihood\_Age, float Liklihood\_Loan,char classVal, float PriorProbability)

{

float Posterior\_Prob = Likelihood\_Age \* Liklihood\_Loan \* PriorProbability;

Console.WriteLine("Posterior Probability " + classVal + ": " + Posterior\_Prob);

return Posterior\_Prob;

}

}

}

**OUTPUT:**

