Machine Learning
(BITS F464)

Assignment 3
Naïve Bayes Classifier

Team Members:

- **Akshay Uppal** **2014A7PS866H**
- **Rounak Rajoria** **2014A7PS857H**
- **Ayush Gupta** **2014A7PS864H**
- **Dushyant Sharma** **2014A7PS861H**

Training using Naïve Bayes Classifier

We have 451 different images in the training data. Each image comprises of 60X70 Pixels. We store the frequency of data for each of 60X70 Pixels in two different matrices Hashgyes and Hashgno .Hashgyes stores number of times a pixel is a hash and it is a face, Hashgno stores number of times the pixel is a hash and it is not a face.

The training data comprises of 150 test images and for each image we calculate the Likelihood of yes and likelihood of no. If likelihood of yes is greater we classify it as a face Else classify it as not a face.

Confusion Matrix

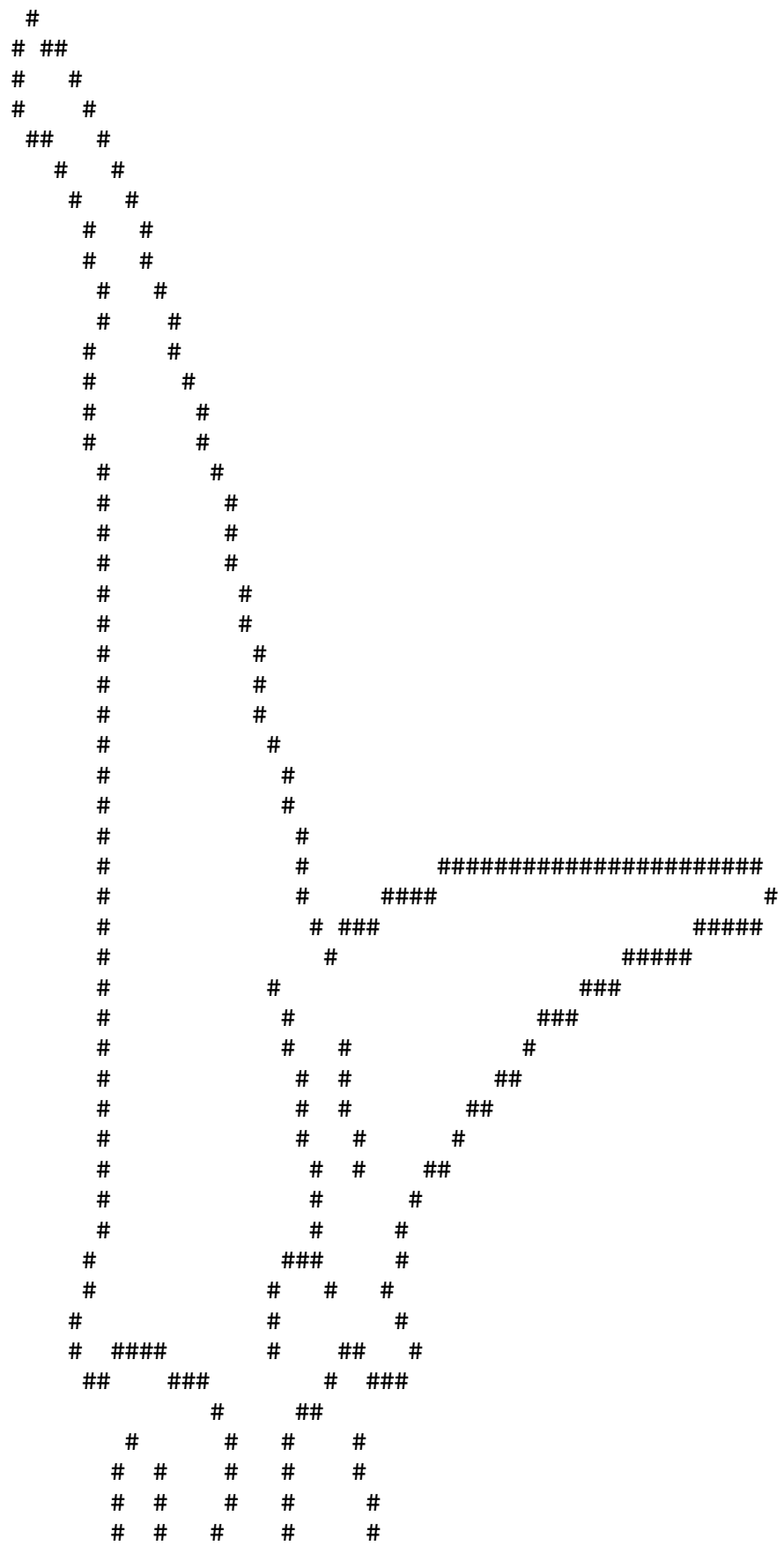
Total =150	**Predicted YES**	**Predicted NO**
Actual YES	**63**	**10**
Actual NO	**7**	**70**

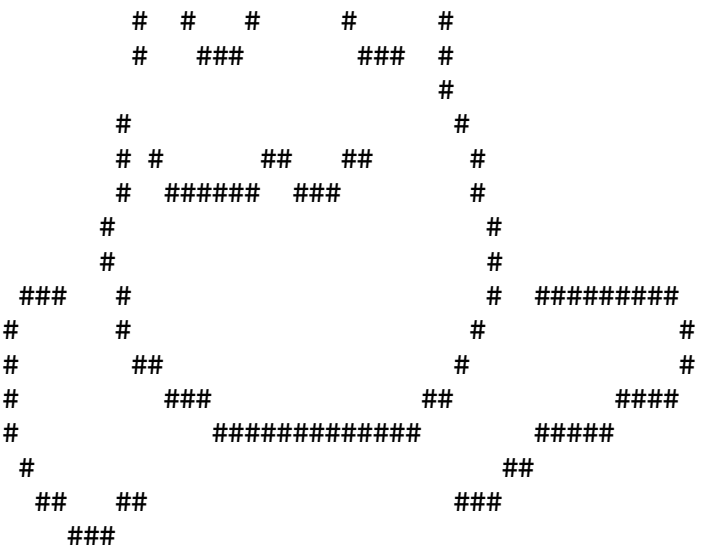
True Positive	63
True Negative	70
False Positive	7
False Negative	10

Accuracy:

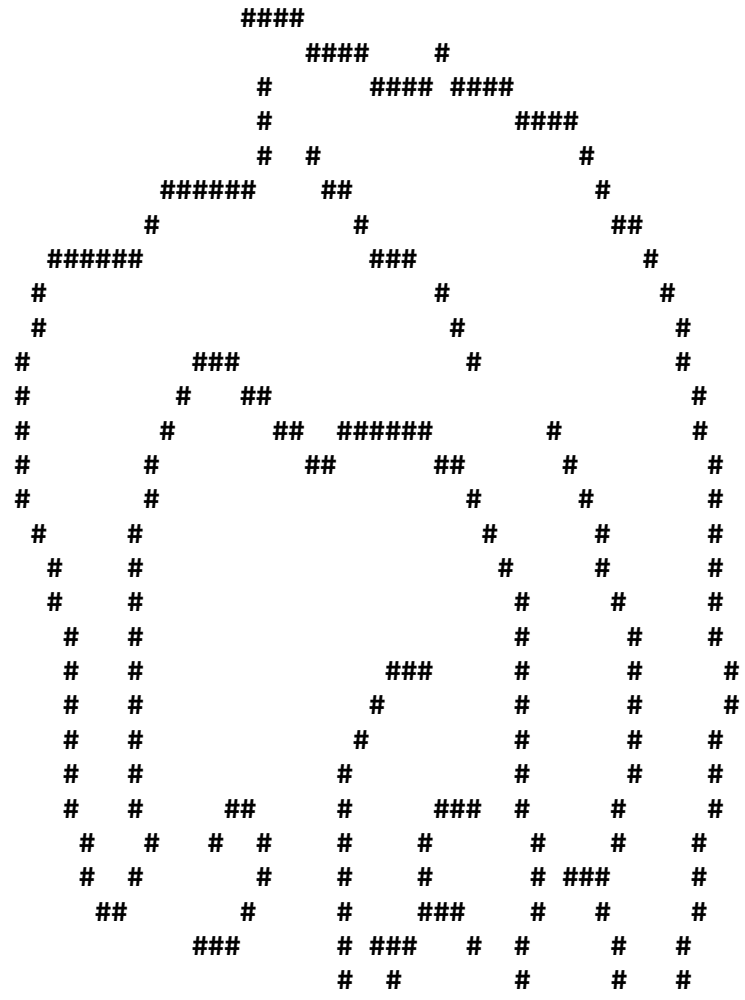
- Test set : 88.67%

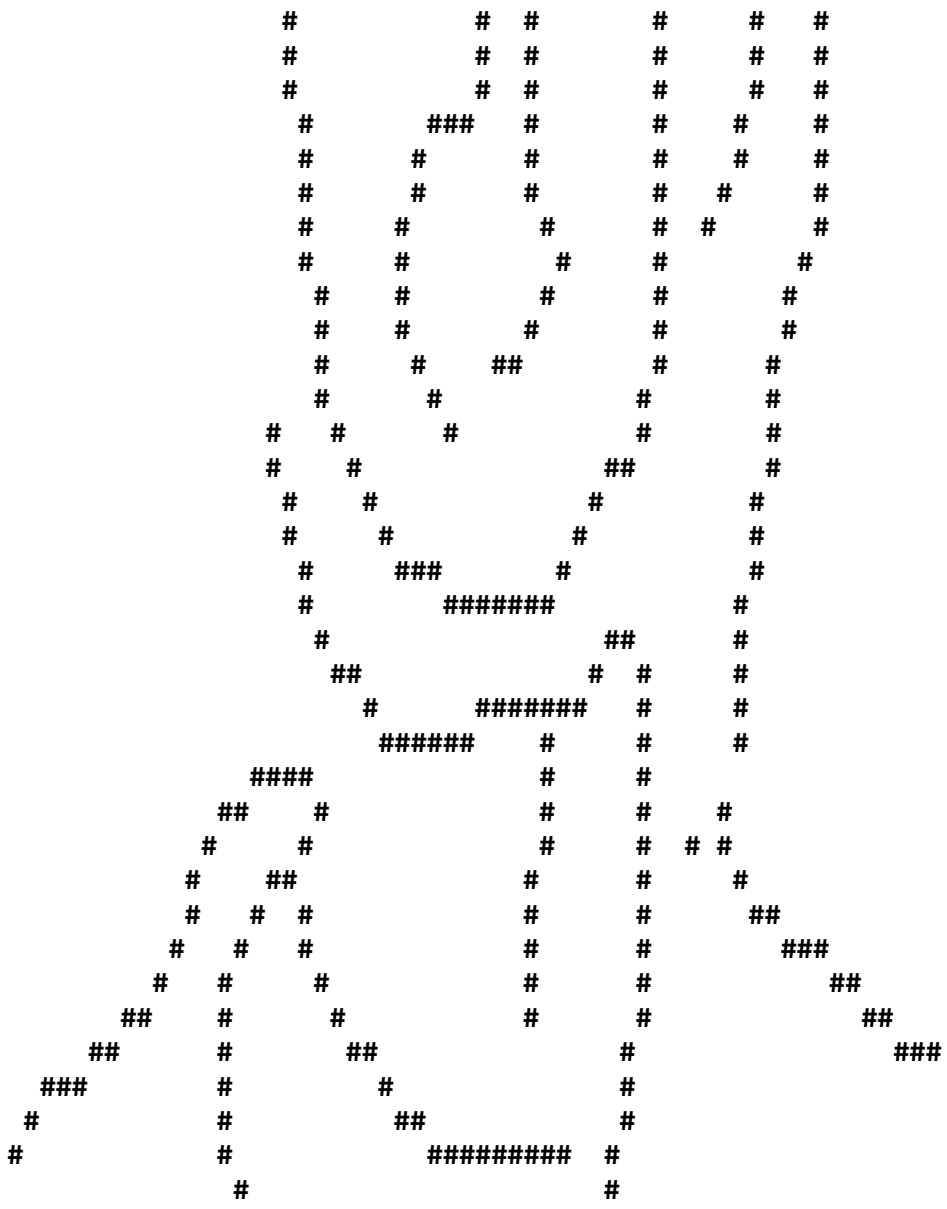
EXAMPLE OF FALSE POSITIVE





Exmaple of False Negative





CODE:

```
#include<bits/stdc++.h>
using namespace std;

int main()
{

int hashgyes[71][61]={0},hashgno[71][61]={0}; //contains count for different pixels if it is hash and the photo is a face and if it is a hash and photo is not a face respectively

long double smooth=1;//Smoothning constant

int n_train=451,n_test=150;//Training and Testing Data
int i,j,k,l,m,n;

int yes=0,no=0;//Counts of yes and no respectively

bool train[451]={0};
FILE*fp=fopen("facedatatrainlabels","r");
int coun=0;
while(coun<n_train)
{
   char c=fgetc(fp);
   if(c=='0'||c=='1')
   {

      if(c=='1')
         {
            train[coun]=1;
            yes++;
         }
         else
         {
            no++;
         }
         coun++;
   }
}
fclose(fp);


fp=fopen("facedatatrain","r");

coun=0;
while(coun<n_train)   //!feof(fp)
{

   for(i=0;i<70;i++)
   {
      for(j=0;j<=60;j++)
      {
         //cout<<i<<" "<<j<<endl;
```

```
            char c=fgetc(fp);
            if(c=='\n')
               {

                  // cout<<"yes"<<endl;

                   break;
               }

            if(train[coun]==1&&c=='#')
            {
               hashgyes[i][j]++; //updating frequency
            }
            if(train[coun]==0&&c=='#')
            {
               hashgno[i][j]++; //updating frequency
            }
         }
      }

coun++;
}
fclose(fp);


//cout<<"hi"<<endl;

bool test[150]={0},ans[150]={0};
int t_yes=0,t_no=0;

fp=fopen("facedatatestlabels","r");
coun=0;
while(coun<n_test)
{
   char c=fgetc(fp);
   if(c=='0'||c=='1')
   {

      if(c=='1')
         {
            test[coun]=1;
            t_yes++;
         }
         else
         {
            t_no++;
         }
         coun++;
   }
}
fclose(fp);
```

```
fp=fopen("facedatatest","r");
coun=0;
while(coun<n_test)   //!feof(fp)
{
   long double pyes=1,pno=1;
   for(i=0;i<70;i++)
   {
      for(j=0;j<=60;j++)
      {
         char c=fgetc(fp);
         if(c=='\n')
            break;
         if(c=='#')
         {
           long double d=hashgyes[i][j]/(long double)yes;
          //  if(d!=0)
               pyes*=(d*smooth);  //Calculating likelihood of yes
            d=hashgno[i][j]/(long double)no;
           //    if(d!=0)
               pno*=(d*smooth);  //Calculating likelihood of no
         }
         else
         {
            long double d=(yes-hashgyes[i][j])/(long double)yes;
           // if(d!=0)
               pyes*=(d*smooth);  //Calculating likelihood of yes
            d=(no-hashgno[i][j])/(long double)no;
            //   if(d!=0)
               pno*=(d*smooth); //Calculating likelihood of no
         }
      }
   }
   pyes*=(yes/(long double)n_train); //Multiplying probability of yes
   pno*=(no/(long double)n_train);  //Multiplying probability of no

   //cout<<"pyes is "<<pyes<<endl;
   //cout<<"pno is "<<pno<<endl;

   if(pyes>pno)
      ans[coun]=1;
   else
      ans[coun]=0;
coun++;
}

int correct=0,tp=0,tn=0,fp1=0,fn=0,fni=-1,fpi=-1;

for(i=0;i<n_test;i++)
{
   if(ans[i]==test[i])
      correct++;
```

```
    if(ans[i]==1&&test[i]==1)
            tp++;
    if(ans[i]==0&&test[i]==0)
            tn++;
    if(ans[i]==0&&test[i]==1)
            {
               fn++;
               fni=i;
               //cout<<"fni is "<<fni<<endl;
            }
    if(ans[i]==1&&test[i]==0)
            {
               fp1++;
               fpi=i;
              // cout<<"fpi is "<<fpi<<endl;
            }
}


//Data for Confusion Matrix

cout<<"True Positive : "<<tp<<endl;
cout<<"False Positive : "<<fp1<<endl;
cout<<"True Negative : "<<tn<<endl;
cout<<"False Negative : "<<fn<<endl;

//Accuracy
cout<<"The accuracy is "<<(correct/(double)n_test)*100<<endl;

}
```